JEUX SUR COMMODORE 64

Jeux d'adresse et de hasard

Philippe FLEURIER
Christian MEILLER

Edimicro

JEUX SUR COMMODORE 64

JEUX D'ADRESSE ET DE HASARD

Collection animée par Benoît de MERLY.

© F.D.S./Edimicro 1984
Première édition

Imprimé en France. Droits mondiaux réservés.

« La loi du 11 mars 1957 n'autorisant, aux termes des alinéas 2 et 3 de l'article 41, d'une part, que les « copies ou reproductions strictement réservées à l'usage privé du copiste et non destinées à une utilisation collective » et, d'autre part, que les analyses et les courtes citations dans un but d'exemple et d'illustration, « toute représentation ou reproduction intégrale, ou partielle, faite sans le consentement de l'auteur ou de ses ayants droit ou ayants cause, est illicite » (alinéa 1er de l'article 40) ».
« Cette représentation ou reproduction, par quelque procédé que ce soit, constituerait une contrefaçon sanctionnée par les articles 425 et suivants du Code pénal ».

ISBN : 2-904457-09-7

EDIMICRO
DÉPARTEMENT ÉDITIONS DE F.D.S. SARL
121-127, avenue d'Italie, 75013 PARIS

JEUX SUR COMMODORE 64

JEUX D'ADRESSE ET DE HASARD

par

Philippe FLEURIER

et

Christian MEILLER

Edimicro

SOMMAIRE

CHAPITRE 1. — **Comment concevoir un programme Basic** 15

1.1 Les idées 17
1.2 Le cahier des charges 19
1.3 L'organigramme 21
1.4 Le codage 24
1.5 La mise au point 25
1.6 Conclusion 26

CHAPITRE 2. — **Techniques de programmation** 27

2.1 La présentation 28
2.2 Possibilités graphiques 30
2.3 Les sprites graphiques 33
2.4 Les possibilités sonores 39
2.5 La rapidité 43
2.6 Contrôles divers 46

CHAPITRE 3. — **Jeux de hasard** 53

3.1 Course d'escargots 53
3.2 Jeu du 21 62
3.3 Balle au pied 68

CHAPITRE 4. — **Jeux d'adresse** 75

4.1 Squash 75
4.2 Casse-briques 83
4.3 Clavier 92
4.4 Trampoline 96
4.5 Soucoupes 104
4.6 Tir sur cible 115
4.7 Poursuite 123
4.8 Course automobile 132
4.9 Puces savantes 140

Annexes 147

A Sprites graphiques 147
B Caractères spéciaux 158

Introduction

Cet ouvrage s'adresse à tous ceux qui désirent à la fois se divertir et s'initier, voire se perfectionner, dans la programmation de jeux en langage BASIC sur le micro-ordinateur personnel "COMMODORE 64".

Il est beaucoup plus agréable d'apprendre en s'amusant, c'est pourquoi le lecteur trouvera dans ce livre, et dans le second volume consacré aux jeux sur COMMODORE 64, un recueil de jeux complets, prêts à l'emploi, faisant appel aux nombreuses possibilités tant graphiques que sonores de l'ordinateur. Ces jeux seront présentés dans plusieurs rubriques distinctes, correspondant à des centres d'intérêts différents: jeux de hasard et jeux d'adresse pour le présent volume.

Dans le deuxième volume, le lecteur trouvera des jeux d'action et des jeux de réfexion.

Ensuite, s'il le désire, le lecteur pourra aller voir de plus près les programmes en eux-mêmes, et tenter d'en saisir le fonctionnement, à l'aide des explications détaillées qui les accompagnent.

C'est dans ce dessein que, dans la suite de cet ouvrage, sera adoptée une présentation en deux parties distinctes:

* Techniques de programmation des jeux en BASIC (et plus particulièrement sur COMMODORE 64).

* Recueil de jeux avec annotations et explications diverses.

Nous espérons que le contenu de ces deux ouvrages vous divertira, et vous donnera envie de programmer à votre tour afin de progresser dans l'acquisition de nouvelles connaissances informatique.

Première partie

Techniques de programmation des jeux

Les techniques qui vont vous être exposées ci-dessous constituent en quelque sorte les bases indispensables à la compréhension des jeux qui vous seront présentés plus loin. Elles vous seront utiles également dans l'élaboration de vos propres jeux.

CHAPITRE **1**

Comment concevoir un programme BASIC

Avant d'aborder tout ce qui concerne les techniques relatives aux jeux, il convient de s'attarder quelque peu sur la façon de créer un programme en langage BASIC.

Cet exposé sera le plus général possible, car, vous le concevez bien, quelle que soit la nature d'un programme (jeu ou non), sa réalisation nécessite le passage par un certain nombre de phases bien définies que l'on a l'habitude de regrouper en cinq blocs qui sont, dans l'ordre:

-Les idées de départ, l'imagination, l'exposé du problème.

-Le cahier des charges, l'ensemble des fonctions à réaliser.

-L'organigramme, l'algorithme de résolution.

-La traduction dans un langage de programmation évolué ou codage.

-La mise au point, les tests de fonctionnement.

Cette méthode de programmation ainsi définie permet d'éviter l'erreur traditionnelle qui consiste à taper le programme directement, sans aucun préliminaire. Ce procédé n'est pas rationnel et ne peut qu'allonger énormément le temps de mise au point, et peut même ne jamais aboutir. Si vous avez la chance que votre programme tourne malgré tout, il se présentera sous forme de "replâtrages" successifs, et il sera impossible de le relire ou de le modifier par la suite.

Nous allons donc examiner ces différentes étapes indispensables qui constituent LA méthode pour réaliser un programme correct, en les axant plus particulièrement sur la conception des jeux.

1.1 Les idées

Le travail de l'imagination constitue la première phase de la conception d'un jeu. Dans un premier temps, toute latitude vous est laissée et vos idées les plus originales, voire saugrenues, pourront être à l'origine de jeux très personnels qui vous divertiront peut-être plus que les "classiques" que vous pourrez trouver partout.

Cependant, avant de vous lancer dans les étapes suivantes, il faut vous demander si le projet que vous avez conçu est réalisable. En effet, malgré toute la liberté dont vous disposez, il restera tout de même quelques contraintes qui feront, ou non, un "véritable jeu" de votre idée de départ.

Mais qu'est-ce qu'un "véritable jeu" ? Voici quelques critères qui pourront vous guider dans votre choix, afin que le jeu soit le plus intéressant possible:

* Le jeu doit éprouver différentes qualités du joueur:

-son intelligence dans les jeux de réflexion,

-ses réflexes dans les jeux d'action,

-sa précision dans les jeux d'adresse,

-sa chance dans les jeux de hasard,

et le faire participer activement par l'intermédiaire du clavier, des joysticks...

* Un jeu doit être équilibré, c'est à dire ni trop facile, ni trop compliqué. En effet, un jeu auquel on gagne ou on perd toujours ne présente pas d'intérêt, et vous ne l'utiliserez jamais. Il faut donc doser la difficulté du jeu pour qu'il capte votre attention le plus longtemps possible, sans vous ennuyer.

Exemple:

supposons que vous vouliez réaliser un jeu de squash. Il vous faut donc: une balle, une raquette, des murs pour y faire rebondir la balle. Ce jeu fait appel à votre habileté et répond tout à fait aux impératifs exposés ci-dessus. On peut également envisager de faire diminuer la taille de la raquette quand le joueur atteint un certain score, accélérer ou ralentir la balle, changer sa direction en fonction du point de la raquette qu'elle frappe...

Nous pousuivrons l'élaboration de ce jeu dans les étapes suivantes.

1.2 Le cahier des charges

Le cahier des charges constitue en quelque sorte le support de votre imagination. Il va falloir y décrire, à partir de vos données primitives, ce que doit effectuer le programme, ses objectifs, et dans le cas d'un jeu, ses règles, sa présentation, la part du joueur et de la machine, la manière dont va s'effectuer le dialogue homme ordinateur...

Tous ces points importants constituent la base de l'étape suivante et doivent être respectés le plus scrupuleusement possible. Nous verrons cependant que, dans l'étape qui suit, il peut arriver que l'on revienne sur le cahier des charges sous le poids de contraintes de "faisabilité": temps d'exécution prévu trop important, difficultés algorithmiques, mauvaises prévisions dans la manière de présenter le jeu...

Exemple:

Nous poursuivons ici le développement de notre jeu de squash, et nous allons en établir le cahier des charges.

- Le jeu consiste à faire rebondir une balle sur les murs d'une pièce à l'aide d'une raquette.

- La raquette doit se déplacer verticalement le long d'une paroi.

- La balle se déplace avec des angles différents suivant l'endroit de la raquette où elle rebondit: au milieu, le rebond est normal et sur les bords, l'angle avec lequel repart la balle est aléatoire (notons que le rebond normal est celui où la balle repart avec le même angle qu'à l'arrivée).

- Si la balle passe derrière la raquette, le joueur a perdu.

- Quand le joueur a atteint un certain score (à déterminer par la suite), la raquette change de dimension, et ceci alternativement (on envisage deux tailles de raquettes).

- Le joueur dispose de trois balles en tout.

- La machine demande le mode de déplacement de la raquette:

    * clavier,
    * joystick (manche à balai).

- Le joueur déplace la raquette avec l'un des modes de déplacement précédents.

- La machine comptabilise le nombre de rebonds sur la raquette pour établir le score final.

- A la fin du jeu, quand l'utilisateur a perdu les trois balles, elle affiche le résultat et demande au joueur s'il désire jouer de nouveau.

1.3 L'organigramme

C'est dans cette phase de la programmation que vous allez concevoir la structure de programme qui "colle" le plus à la description du jeu rédigée dans la partie concernant le cahier des charges. Cet organigramme pourra se présenter sous la forme d'un dessin ou d'une suite d'instructions structurées de façon à faire apparaître clairement la construction du programme.

Il faut aussi y définir les blocs qui vont correspondre à des fonctions élémentaires du programme. Cette structure par blocs distincts permet une meilleure compréhension et une mise au point de chaque partie indépendamment des autres, ce qui est nettement plus efficace pour repérer les erreurs éventuelles !

Le découpage en blocs présenté au-dessus ne correspond pas nécessairement à l'utilisation de sous-programmes, mais surtout à une séparation des différentes unités logiques de l'ensemble. Cependant, il faut reconnaître que, souvent, cette structure fera largement appel aux sous-programmes comme vous pourrez le constater dans les jeux qui constituent ce recueil.

L'effort de structuration du programme peut malgré tout s'avérer difficile, voire même impossible, pour des raisons déjà évoquées dans le paragraphe traitant du cahier des charges: temps d'exécution prévu trop important (ceci est un point fondamental pour les jeux d'action), trop

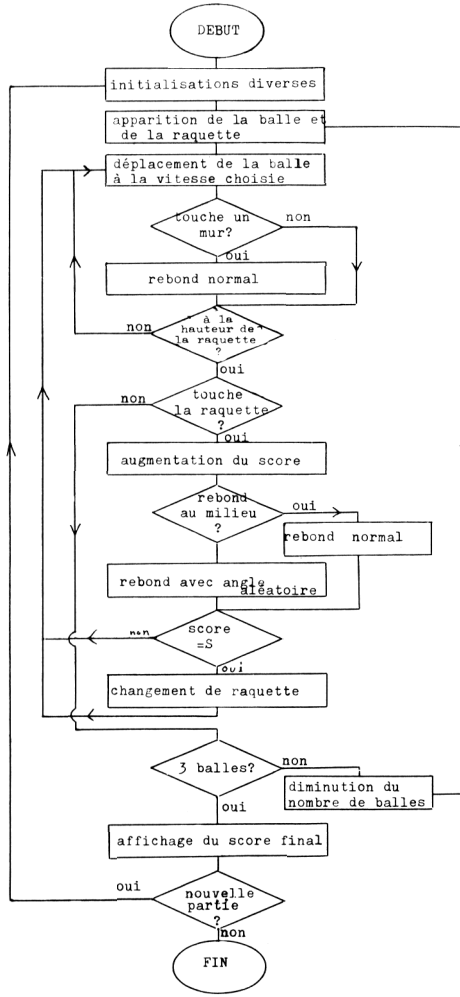
grande complexité de l'algorithme, longueur du programme (la capacité mémoire est une ressource critique)... C'est pourquoi vous pouvez être amené à revenir sur le cahier des charges pour en modifier éventuellement certains termes, afin de rendre le programme réalisable selon les critères que vous avez pu définir lors de votre première tentative. Nous allons voir sur notre exemple comment dessiner un organigramme:

Exemple:

Nous allons ici présenter l'organigramme du jeu de squash que nous nous proposons de réaliser.

L'organigramme général d'un programme comporte en fait plusieurs sous-organigrammes correspondant chacun à un bloc logique du jeu. Dans cet exemple, nous allons dessiner l'organigramme du programme principal de notre jeu de squash. Nous vous laissons le soin de réaliser, si le coeur vous en dit, celui de chaque sous-partie du programme.

Dans la rubrique "Initialisations diverses", nous regroupons les valeurs initiales des variables, définissons les sprites, les couleurs du fond et du cadre, etc... S représente la valeur du score pour laquelle la raquette change de dimension (S est modifié quand la raquette change de façon à obtenir l'alternance des deux raquettes).


DEBUT
initialisations diverses
apparition de la balle et de la raquette
déplacement de la balle à la vitesse choisie
touche un mur?
non
oui
rebond normal
à la hauteur de la raquette ?
non
oui
touche la raquette ?
non
oui
augmentation du score
rebond au milieu ?
oui
rebond normal
rebond avec angle aléatoire
score =S
non
oui
changement de raquette
3 balles?
non
diminution du nombre de balles
oui
affichage du score final
nouvelle partie ?
oui
non
FIN

1.4 Le codage

Quand l'organigramme a été dessiné ou l'algorithme établi de façon précise, et que la logique de chaque bloc constitutif du programme a été éprouvée, il est ensuite possible de passer à la traduction dans un langage de programmation évolué, en l'occurence le langage BASIC, que votre micro-ordinateur est capable de comprendre.

Ce n'est qu'à partir de cette étape que des connaissances spécifiques sont indispensables. Ce sont, en fait, les mots et la syntaxe du langage que vous devez maîtriser pour être capable de traduire sans difficultés l'algorithme que vous avez conçu. Il est important de remarquer que les étapes précédentes ne faisaient en aucun cas appel à des connaissances préalables en informatique, et que, seul, un minimum de bon sens était indispensable.

Là encore, il faut adopter une démarche rationnelle pour procéder au codage. Par exemple, dans la numérotation des lignes du programme, il est (très fortement) conseillé de progresser de 10 en 10, afin de pouvoir insérer des lignes lors de la mise au point sans avoir à déplacer des blocs entiers de programme, ce qui est excessivement fastidieux et peut être à l'origine de nouvelles erreurs. Il convient également de numéroter chaque bloc logique (sous-programme ou non) afin qu'il se détache le plus possible des autres, ce qui, vous vous en rendrez compte, facilite la relecture du programme entier lors d'une modification éventuelle: de 1000 en 1000 par exemple.

Exemple

Vous pourrez vous reporter au jeu de squash qui figure dans le recueil au paragraphe 2.1 qui est celui consacré au jeu de squash.

1.5 La mise au point

En théorie, cette étape est superflue, et le programme devrait pouvoir fonctionner sans passer par une mise au point. Malheureusement, dès que sa longueur augmente un peu, un programme contient immanquablement des erreurs qui peuvent être de plusieurs types.

Les erreurs d'algorithme sont les plus graves. En effet, dans ce cas, il faut revenir sur les deux étapes précédentes, voire même sur la première. En général, ces erreurs sont dûes à une mauvaise analyse du problème, et un programmeur attentif ne devrait pas se trouver devant ce type de problème.

Les erreurs communes se situent le plus souvent dans le passage de l'organigramme au codage. La meilleure façon de les rectifier consiste à tester l'exécution du programme bloc par bloc, d'où l'intérêt d'avoir structuré le programme de façon modulaire et d'avoir adopté une numérotation rationnelle. Il vous suffit alors, par exemple, d'intercaler une ligne avec une instruction STOP

ou une instruction END à la fin du bloc et de taper au clavier RUN N, où N est le numéro de la première ligne du bloc considéré.

1.6 Conclusion

Même si la méthode qui vous a été exposée peut paraître quelque peu contraignante, voire même relativement lourde, vous vous rendrez vite compte qu'au bout de peu de temps, vous programmerez très facilement de beaux jeux, aidé des quelques conseils qui vont faire l'objet du second chapitre de cette première partie. Vous ne perdrez pas de temps en "rafistolages" et réaliserez des programmes complexes, lisibles, efficaces et sans erreurs.

CHAPITRE **2**

Techniques de programmation

La méthode exposée dans le chapitre précédent est nécessaire à la réalisation d'un programme de jeu. Cependant, elle n'est pas suffisante pour considérer que le jeu qui a été créé est beau, attrayant, original, etc...

Les micro-ordinateurs possèdent maintenant de nombreuses possibilités qu'il faut utiliser au mieux pour rendre un jeu intéressant. Nous allons donc exposer un certain nombre de techniques, les unes d'ordre général, les autres s'appliquant directement à votre COMMODORE 64.

2.1 La présentation

Le terme présentation regroupe plusieurs points importants dans la programmation: l'aspect extérieur du jeu, les couleurs, les sons, le texte, etc...

En ce qui concerne le texte, il est indispensable de présenter le jeu par son titre avant de le faire démarrer. Ce titre peut être accompagné d'un mode d'emploi le plus succint possible, bien utile quand l'utilisateur n'est pas un habitué du jeu. Il faut préciser également comment le joueur va communiquer avec la machine: joystick, clavier (et dans ce cas, quelles touches utiliser), etc...

Durant le déroulement du jeu, il est aussi possible d'indiquer le score réalisé par le joueur, détail parfois bien utile pour connaître la stratégie à adopter ensuite. Enfin, quand le jeu se termine, il est indispensable d'afficher à l'écran le score final, car un jeu sans score est un jeu sans intérêt. Comment évaluer un score? Il suffit d'avoir un peu d'imagination et de logique: ajouter des points quand le joueur atteint un vaisseau ennemi, ou détruit un de ses missiles, ou lorsqu'il dévore une proie, double un concurrent dans une course, etc... Soustraire quand le vaisseau tire un missile, quand le carburant s'épuise, etc...

Le dernier détail à introduire à la fin de vos programmes est de demander à l'utilisateur s'il

désire jouer de nouveau. Vous vous apercevrez assez vite que cette possibilité de votre programme évite les manipulations fastidieuses, car vous utiliserez souvent votre jeu plusieurs fois de suite, ne serait-ce que pour améliorer votre score...

Mis à part les différents textes de présentation, il faut également se préoccuper de l'esthétique qui, bien travaillée, rend le jeu nettement plus attrayant. Le programmeur peut ainsi améliorer sensiblement la qualité graphique -formes et couleurs- ainsi que la sonorité. Tout ceci est indispensable, et la tâche vous est facilitée car vous disposez d'un micro-ordinateur très performant dans ces domaines. Ces efforts de présentation peuvent être lourds et même représenter plus de travail que pour la conception de la partie exécution du programme. Mais l'intérêt que l'on porte à un jeu tient souvent à son aspect (et l'on pourrait même ajouter qu'il y est directement proportionnel). En outre, cette présentation vous est personnelle, et c'est un point fondamental dans toute création. Nous allons voir plus loin la manière d'utiliser les possibilités sonores et graphiques de votre COMMODORE 64.

2.2 Possibilités graphiques

Notre propos n'est pas ici de développer tout ce qui concerne les possibilités graphiques de votre appareil, non plus que leur utilisation, que vous trouverez dans n'importe quel manuel d'utilisation du COMMODORE 64. Il s'agit plutôt de rappeler quelques points indispensables à la création des jeux.

Le COMMODORE 64, en version de base, dispose d'un graphisme basse résolution de 40*25 points, et d'un jeu de caractères graphiques de haute qualité, tout cela agrémenté d'un ensemble de 16 couleurs possibles pour tous ces caractères. Malgré tout, la basse résolution donne un graphisme relativement figé (ce détail est compensé par la grande variété des caractères et des couleurs).

Pour visualiser à l'écran un caractère graphique, deux possibilités s'offrent à vous: soit imprimer ce caractère à l'aide de l'instruction "PRINT", soit mettre le code de ce caractère dans la case mémoire d'écran correspondante.

Dans le premier cas, la forme de l'instruction de base à exécuter est simple. Il vous suffit de taper, directement ou dans un programme, PRINT"C", où C est le caractère graphique que vous obtenez en appuyant par exemple simultanément sur les touches SHIFT et C (C apparaît sur le devant des touches). Imprimer le caractère à l'endroit où se

trouvait le curseur précédemment n'est pas suffisant, il faut pouvoir le positionner à une place voulue. Vous disposez pour cela des touches de déplacements verticaux et horizontaux (CRSR), et vous allez donc taper:

PRINT « Q Q Q ◢ »

ce qui va imprimer votre caractère à la place donnée par les déplacements, mais à partir de la position où se trouvait le curseur précédemment. Ainsi, pour être sûr que, quelle que soit la place précédente du curseur, le caractère se trouve au même endroit, il suffit de taper avant une instruction qui ramène le curseur en haut à gauche de l'écran, soit: (il s'agit de l'instruction CLR/HOME).

PRINT « S »

La couleur du caractère graphique est variable, et est obtenue en tapant sur la touche de contrôle (CRTL) et sur un chiffre de 1 à 8 simultanément, ou bien sur la touche Commodore (C=) et sur un chiffre de 1 à 8. Vous disposez ainsi de 16 couleurs possibles. En combinant les diverses possibilités exposées au-dessus, ainsi que celle offerte par la vidéo inverse (CRTL 9: ON et CRTL 0: OFF), il est alors possible de concevoir la réalisation d'un décor pour un programme de jeu. Pratiquement, vous allez taper par exemple:

PRINT « ♡ E A »

La seconde méthode de gestion du graphisme consiste à utiliser directement la mémoire d'écran par l'intermédiaire de l'instruction POKE. Dans la

mémoire de votre ordinateur, l'écran de votre téléviseur se présente sous la forme de deux grilles de 25*40 cases chacune, représentant toutes les positions possibles d'un caractère graphique sur l'écran. La première de ces grilles correspond au caractère graphique: en effet, la case mémoire portant ce numéro contient le code du caractère considéré. On affiche donc le caractère voulu à l'endroit voulu en tapant POKE NU, CO, où NU est le numéro de la case et CO le code du caractère. La valeur de NU est comprise entre 1024 et 2023.

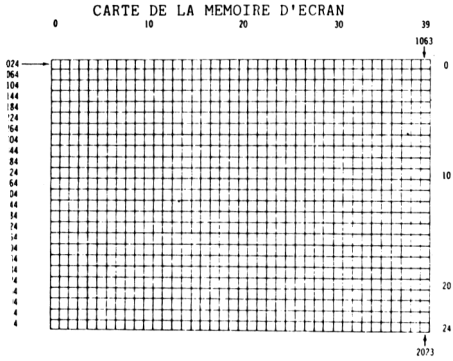
La deuxième grille est affectée à la couleur du caractère, et elle s'utilise de la même façon que la précédente: POKE NU, COULEUR. NU varie dans ce cas entre 55296 et 56285 et COULEUR entre 0 et 15.

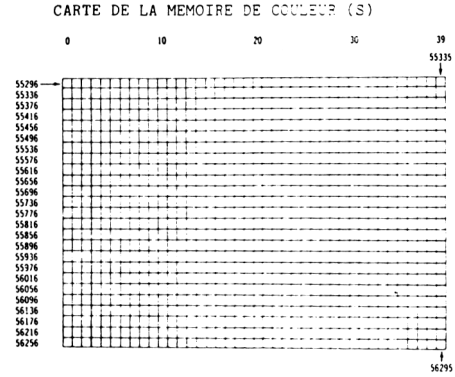
Voyons sur un exemple comment on utilise ces instructions:

```
POKE 1188, 90

POKE 55460, 2
```

Dans cet exemple, nous avons placé la valeur 90 (code du carreau) dans la case mémoire numéro 1188 et la valeur 2 (code de la couleur rouge) dans la case mémoire numéro 55460 Ces deux cases représentant le même point sur l'écran. vous allez donc voir apparaître un carreau de couleur rouge dans le coin supérieur gauche.

Les deux grilles apparaissent ci-dessous.





2.3 Les sprites graphiques

Pour tout ce qui concerne plus particulièrement l'animation des jeux, vous disposez d'une autre possibilité constituée des huit sprites graphiques. Les sprites sont des objets que vous définissez vous-même et qui sont en fait des matrices 24*21 de points en haute résolution (à titre indicatif, l'écran possède 40*25 points en basse résolution et 320*200 points en haute résolution), qualité bien intéressante dans l'élaboration d'objets nouveaux, demandant un graphisme très détaillé. Il est possible d'utiliser jusqu'à huit sprites différents simultanément sur l'écran, mais vous pouvez en définir beaucoup plus en mémoire. De plus, ils sont très faciles à manipuler: déplacements, couleurs, tailles, détection de collisions entre

sprites, ou entre sprites et caractères graphiques, etc...

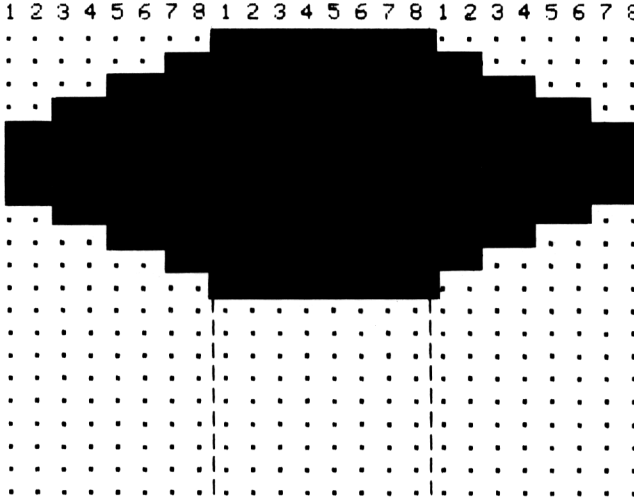
Les sprites ont la particularité d'être indépendants de la page graphique de votre écran. En effet, lorsque vous effacez l'écran, avec l'instruction CLR/HOME, les sprites ne sont pas affectés. En mémoire, le sprite est stocké dans un bloc de 21*24 cases que l'on doit donc remplir par des valeurs qui vont correspondre à l'objet qu l'on désire construire.

Le meilleur moyen pour exposer la création d'un sprite est de travailler sur un exemple: nous allons donc créer un sprite qui représentera une soucoupe volante. Tout d'abord, il faut dessiner une grille de 21*24 cases, et remplir les cases avec le dessin. Jusque là, rien de compliqué.

Il vous faut ensuite numéroter les lignes et les colonnes de votre grille de la façon indiquée sur notre grille exemple. Ceci va vous servir à convertir l'information contenue dans la grille en une information compréhensible par l'ordinateur, sous forme de nombres dont la donnée correspondra exactement au dessin initial.

La numérotation se fait de 1 à 21 pour les lignes, mais attention, pour les colonnes elle est tout à fait particulière. Les colonnes vont être divisées en trois groupes de 8, numérotées de la même façon: de gauche à droite, on va écrire les nombres 128, 64, 32, 16, 8, 4, 2, 1.

Lorsque tout cela est fait, on peut considérer que le travail préliminaire est terminé. Déterminons maintenant les nombres que nous allons effectivement donner à la machine. En face de chaque ligne de cases, comme sur la grille exemple, vous allez obtenir trois nombres différents. Le premier se calcule en additionnant les nombres (128,64,...,1) correspondant aux cases remplies sur le dessin. Les autres se calculent de la même façon, et pour toute la figure, la machine a donc besoin de 63 nombres (notons que lorsqu'une case est vide, le nombre associé est évidemment nul). Vous pouvez à présent regarder l'exemple ci-dessous pour mieux comprendre comment se construit un sprite.



1	0	255	0
2	3	255	192
3	15	255	240
4	127	255	252
5	255	255	255
6	255	255	255
7	255	255	255
8	255	255	255
9	127	255	252
10	15	255	240
11	3	255	192
12	0	255	0
13	0	0	0
14	0	0	0
15	0	0	0
16	0	0	0
17	0	0	0
18	0	0	0
19	0	0	0
20	0	0	0
21	0	0	0

Votre objet est alors créé sur le papier, il reste à le mettre dans la mémoire de votre COMMODORE 64. La mémoire des sprites est organisée en blocs qui vont contenir chacun une grille telle qu'elle a été définie au-dessus. Les adresses de ces blocs sont contenues dans 8 emplacements mémoire, numérotés de 2040 à 2047, qui correspondent aux sprites numérotés de 0 à 7. Si

vous mettez la valeur 13 à l'adresse 2043, cela voudra dire que le sprite 3 va trouver ses données dans le bloc 13, c'est-à-dire de l'adresse 832 (13*64) à l'adresse 895 (13*64+63). Cependant, il faut faire très attention au nombre de sprites que vous désirez définir, car sans exécuter d'instructions spéciales, il n'est pas possible de définir plus de quatre sprites à la fois dans les blocs suivant le bloc 13 (seuls les blocs 13, 14 et 15 sont possibles dans ce cas, car 16*64=1024 est une adresse de l'écran; voir le paragraphe sur le graphisme. Quand vous utiliserez plus de quatre sprites différents, il vous faudra donc exécuter la séquence suivante avant le chargement en mémoire de votre programme:

POKE 44,16 :POKE 16*256,0 :NEW et return.

Vous pouvez donc, après avoir tapé ces ordres, utiliser les blocs portant les numéros 32 à 63.

Les nombres ainsi obtenus sont tapés dans votre programme, dans des lignes de données (DATA), selon l'ordre naturel: d'abord les trois premiers nombres de la première ligne, puis ceux de la deuxième, et ainsi de suite.

Dans toutes les instructions de manipulation des sprites, vous allez devoir vous habituer à ne pas donner directement les numéros des sprites, mais un autre nombre en fait égal à 2 exposant le numéro du sprite: soit, pour le sprite 4 la valeur (2 exposant 4) =16.

Ainsi, lorsque vous désirez faire apparaître les sprites 2 et 5 à l'écran simultanément, vous exécutez l'instruction:

```
POKE C+21,4+32
```

où C prend la valeur 53248.

Le déplacement des sprites est effectué par le même type d'instructions:

abscisse du sprite numéro n: registre C+2*n,

ordonnée du sprite numéro n: registre C+2*n.

Ainsi, pour faire apparaître le sprite numéro 4 à l'emplacement repéré par le point X,Y, il suffit de faire:

```
POKE C+6, X: POKE C+7, Y
```

Il convient de noter que le point repéré par X,Y se trouve en haut à gauche de la matrice 21*24 constitutive du sprite.

Mais, attention, les valeurs de X et Y doivent être comprises entre 0 et 255. Pour l'ordonnée Y, cela ne pose pas de problème, car l'écran ne comporte que 200 points de hauteur. Pour l'abscisse X, le problème est différent, car

l'écran est composé de 320 points en largeur. Afin de pouvoir déplacer les sprites au-delà de l'abscisse 255, il suffit de mettre dans le registre d'adresse C+16 le nombre correspondant aux sprites qui se trouveraient dans cette partie de l'écran, soit:

```
POKE C+16, 1+16+128
```

si ce sont les sprites 0, 4 et 7 qui sont dans ce cas.

Les couleurs des sprites peuvent aussi être choisies parmi les 15 couleurs possibles (que nous avons évoquées précédemment). Là encore, il vous est facile de changer à volonté ces couleurs en mettant leur numéro dans un registre déterminé. Pour les sprites de 0 à 7, ces registres portent les numéros C+39, C+40,...,C+46. Ainsi, pour donner au sprite 6 la couleur bleue, vous devez taper:

```
POKE C+45,6
```

Vous disposez encore d'une autre possibilité de manipulation des sprites qui consiste à doubler leur taille en largeur et en hauteur, de façon indépendante. Voici un exemple dans lequel nous doublons la taille en largeur des sprites 0 et 5, en hauteur des sprites 2 et 5:

```
POKE C+29,1+32

POKE C+23,4+32
```

le sprite 5 va doubler sa taille dans toutes les directions.

En fait, on utilisera le plus souvent les caractères graphiques et leurs nombreuses possibilités de couleurs pour créer le fond, le paysage, les décors des jeux. Les sprites seront alors réservés à l'animation, aux objets à déplacer, et à "fignoler" dans leur présentation.

Dans les annexes qui se trouvent à la fin de cet ouvrage, vous pourrez trouver l'ensemble des sprites définis dans les programmes de jeu.

2.4 Les possibilités sonores

Les possibilités sonores du COMMODORE 64 sont très étendues et permettent une variété quasi illimitée de combinaisons. Si vous avez envie d'obtenir des précisions sur la programmation des sons, vous pourrez vous reporter à un manuel d'utilisation.

Il faut cependant savoir que l'on peut régler l'attaque/chute, le soutien/relâche de toutes les notes, ainsi que le volume, la modulation, sur une plage de huit octaves et ceci pour trois voix indépendantes.

Le volume peut se régler sur une échelle qui varie de 0 à 15, mais, en pratique, vous n'utiliserez que le maximum afin de pouvoir entendre quelque chose (le maximum est 15).

Pour définir un son, il est indispensable de définir la modulation sous laquelle il va être émis. A titre indicatif, la modulation correspond en fait à la forme du son, et l'on peut prendre les exemples suivants:

* Modulation TRIANGLE: Flûte, Xylophone, Orgue, Accordéon...

* Modulation DENTS DE SCIE: Clavecin, Banjo...

* Modulation PULSATION: Piano, Moteur...

* Modulation BRUIT BLANC: Explosions, Tirs, Vagues de la mer...

Vous trouverez ci-dessous le tableau des valeurs à taper pour obtenir la modulation que vous désirez pour la voix choisie. La forme de l'instruction est la suivante:

POKE ADRESSE,VALEUR

Notez que la première valeur (17, 33, 65, 129) amorce la modulation et que la seconde (16, 32, 64, 128) l'arrête.

CONTROLE DE DEMARAGE ET D'ARRET DES NOTES					
	MODULATION	TRIANGLE	DENT DE SCIE	PULSATIONS	BRUIT BLANC
VOIX 1	54276	17/16	33/32	65/64	128/128
VOIX 2	54283	17/16	33/32	65/64	128/128
VOIX 3	54290	17/16	33/32	65/64	128/128

Nous avons dit plus haut que le réglage de l'attaque/chute était possible pour chaque note. Mais peut-être faut-il préciser ce que cache ce terme. L'attaque représente la vitesse de montée d'une note à son niveau de volume maximum. La

chute d'une note, au contraire, est sa vitesse de retour du volume maximal au volume de soutien.

Vous avez la possibilité d'ajuster ces deux valeurs dans une seule opération, qui est la suivante:

POKE ADRESSE,NOMBRE

où NOMBRE est choisi parmi toutes les valeurs possibles regroupées dans le tableau qui suit.

REGLAGE DES TAUX D'ATTAQUE/CHUTES

	CONTROLE D'ATTAQUE/ CHUTE	ATTAQUES FORTE	MOYE	FAIB	TRES FAIB	CHUTES FORTE	MOYE	FAIB	TRES FAIB
VOIX 1	54277	128	64	32	16	8	4	2	1
VOIX 2	54284	128	64	32	16	8	4	2	1
VOIX 3	54291	128	64	32	16	8	4	2	1

Après le réglage de l'attaque/chute, vient celui de soutien/relâche. Le soutien représente le temps pendant lequel la note va rester à son volume normal, et le contrôle de relâche correspond à la vitesse de disparition de la note après la phase de soutient.

REGLAGE DU TAUX SOUTIEN/RELACHE

	CONTROLE DE SOUTIEN/ RELACHE	SOUTIEN FORTE	MOYE	FAIB	TRES FAIB	RELACHE FORTE	MOYE	FAIB	TRES FAIB
VOIX 1	54278	128	64	32	16	8	4	2	1
VOIX 2	54285	128	64	32	16	8	4	2	1
VOIX 3	54292	128	64	32	16	8	4	2	1

Lorsque tout ce qui précède a été fixé, il ne reste qu'à déterminer quelle est la note qui va être jouée dans la gamme. La plage de variation est relativement importante, puisque les notes

accessibles s'étalent sur 7 octaves. Pratiquement, il est nécessaire de disposer de deux valeurs pour une seule note (les valeurs de haute et basse fréquences de la note considérée).

Des tableaux de toutes ces valeurs sont fournis dans le manuel qui est livré avec votre ordinateur.

Mais notre propos, dans ce paragraphe, consiste plutôt à indiquer que les sons sont indispensables dans presque tous les jeux. Dans les jeux d'action, on les utilise pour rythmer le déplacement des objets animés, pour simuler les tirs, les explosions, etc... Dans les jeux de réflexion, ils servent aussi à rompre un silence qui peut vite devenir ennuyeux pour l'utilisateur.

Dans tous les cas, le son est un moyen supplémentaire d'animation qui apporte un nouvel intérêt au jeu. Il faudra donc utiliser au mieux, c'est-à-dire suivant l'effet que vous voulez obtenir, les possibilités sonores de votre ordinateur. Pour cela, le meilleur moyen consiste à essayer différentes combinaisons jusqu'à ce que l'on obtienne entière satisfaction sur la nature du son désiré.

2.5 La rapidité

Nous abordons ici un point critique de la programmation des jeux, et plus particulièrement celle des jeux d'action. Le temps d'exécution d'un programme est la plupart du temps un problème qui se pose rapidement au programmeur. Le langage BASIC est un langage interprété, ce qui signifie que chaque ligne du programme est traduite par l'ordinateur dans son langage machine, le seul qu'il soit capable de comprendre, au fur et à mesure que ce dernier la rencontre, afin de l'exécuter ensuite. Il existe une autre façon de procéder, qui consiste à traduire une bonne fois pour toutes le programme au début et de le faire démarrer ensuite: c'est la compilation.

Vous n'aurez aucune difficulté si le programme est court. Malheureusement, un jeu tant soit peu élaboré devient souvent trop lent pour être un vrai jeu d'action. Pour résoudre ce problème, une solution radicale serait de programmer en assembleur, ou en langage machine. Mais ce mode de programmation n'est pas très aisé. Rassurez-vous, il est tout à fait possible de réaliser de très beaux jeux, suffisamment rapides, moyennant quelques astuces et habiles techniques, qui feront gagner du temps à l'exécution de votre programme.

Le premier point important est de placer la "boucle programme", c'est-à-dire la séquence d'instructions qui se répète tout au long du déroulement du jeu, au début de la liste des instructions. Il est également conseillé de réduire le plus possible la longueur de la boucle

en groupant les tests, en utilisant des sous-programmes pour en alléger l'écriture, en ne sortant de cette boucle que sur intervention extérieure de l'utilisateur, etc...

En règle générale, il faut s'efforcer de faire les programmes le plus court possible, car la longueur totale influe sur la vitesse d'exécution. Voyons quelques moyens de raccourcir les programmes:

* Supression des lignes de commentaire,

* Tassement des lignes par écriture de plusieurs instructions par ligne,

* Utilisation de variables indépendantes au lieu de tableaux indicés,

* Utilisation de variables au lieu de constantes quand une valeur revient un certain nombre de fois dans un programme,

* Lorsque vous utilisez plusieurs tests en cascade, il est plus rapide, quand c'est possible, d'employer les instructions ON GOTO ou ON GOSUB.

* Suppression de la variable de boucle après le NEXT de fin (FOR I=1 to 10:.....: NEXT). En effet, si l'on avait écrit NEXT I, l'ordinateur aurait vérifié qu'il s'agissait bien de I et cela utilise un certain temps. (Il faut cependant noter qu'il n'est pas toujours possible de supprimer cette variable, notamment quand il y a des enchevêtrements de boucles, et que la machine a du mal à s'y retrouver !)

Le déplacement des objets doit donner une impression de continuité, il convient donc - en théorie - de mouvoir les mobiles point par point de façon à obtenir le meilleur effet possible. Mais, dans ce cas, le jeu restera souvent désespérément lent dans le déplacement de ces objets. Il ne vous reste alors qu'une seule possibilité, c'est de faire se déplacer les objets de plusieurs points à la fois, c'est-à-dire par saccades, mais de façon raisonnable, car des sauts trop larges feront perdre au jeu son réalisme et donc son intérêt. Là encore, il s'agit d'une question de dosage, et le meilleur moyen de l'effectuer est d'essayer plusieurs valeurs et de constater "de visu" le résultat.

Toutes ces possibilités pour rendre vos programmes plus rapides ne représentent cependant que des gains de temps qui ne seront en aucune façon comparables à ceux que vous pourrez obtenir en réalisant des jeux bien construits et bien analysés grâce à la méthode qui vous a été présentée dans le premier chapitre.

Pour terminer ce paragraphe, il faut préciser que même dans les jeux de réflexion, le facteur temps est critique. Moins que dans les jeux d'action, c'est évident. Mais il est très désagréable, même si le programme est très efficace et très bien construit, de jouer à un jeu de réflexion qui demande un temps de réponse de plus de 10 minutes !!!

Enfin, si toutes vos tentatives pour rendre le jeu plus rapide restent vaines, il faudra vous résoudre soit à le programmer en langage machine, soit à compiler le programme BASIC que vous aurez

déjà réalisé. La seconde méthode est plus simple car elle ne demande aucune connaissance autre que le BASIC, bien qu'exigeant la possession d'un compilateur sur disquette par exemple.

2.6 Contrôles divers

Il est bon d'insister sur le fait que le programme doit être armé contre les manipulations illicites ainsi que contre les déplacements des mobiles en dehors de l'écran. Toute les données que vous allez entrer au clavier sur la demande de votre appareil peuvent être cause d'erreurs qui vont soit "planter" votre programme, et il vous faudra alors le faire redémarrer, soit introduire des valeurs aberrantes dans les variables utilisées, et votre jeu fera n'importe quoi. Dans tous les cas, ces erreurs peuvent être évitées en contrôlant de façon précise chaque nouvelle entrée de variable, et chaque déplacement du mobile: il faut minimiser les risques en protégeant les programmes.

Pour éviter tous ces désagréments, il vous suffit, lors de la saisie des données au clavier, de tester les variables suivant leur nature:

* Les lettres ont une valeur nulle, vous pouvez donc les repérer en utilisant la fonction VAL,

* Vous pouvez également tester si la variable est entière ou non en utilisant la fonction INT, partie entière, la partie décimale sera donc le

nombre moins sa partie entière,

* Il ne faut pas oublier non plus, après toutes les vérifications précédentes, de tester si la variable est dans l'intervalle voulu,

* Vous devez vous assurer que vos valeurs ne vont pas dépasser la capacité de stockage de la machine dans une variable ou que vos chaînes de caractères ne vont pas être trop longues.

Mis à part ces tests qui concernent les variables entrées au clavier, il convient de tester si les mobiles qui forment l'animation de votre jeu ne sortent pas de l'écran. Bien sûr, les tests vont se rajouter aux autres et vont représenter une perte de temps certaine dans l'exécution de votre programme, mais il vaut mieux perdre un peu de rapidité au profit d'un programme qui ne se "plante"pas.

Deuxième partie

Recueil de jeux en BASIC

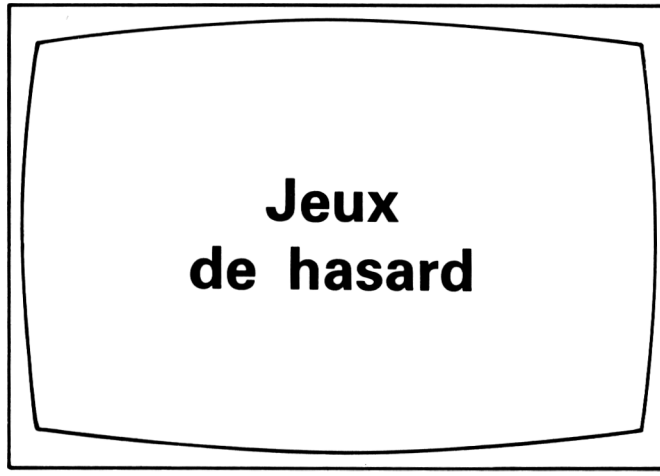
Les jeux en BASIC qui constituent ce recueil sont présentés dans deux chapitres qui regroupent des jeux de hasard et des jeux d'adresse Ces jeux respectent la méthode de programmation et les diverses techniques exposées dans la première partie de cet ouvrage. ils pourront donc vous servir d'exemples d'application.

La plupart des jeux contiendront un mode d'emploi succinct que vous pourrez supprimer en tapant le programme, mais rappelez-vous que si vous jouez à un jeu que vous n'avez pas utilisé depuis longtemps, il se peut que vous ayez oublié ses règles, ou le mode de déplacement des objets, ou bien encore sur quelles touches appuyer pour pouvoir répondre à la machine.

Les programmes font appel, pour leur utilisation, au clavier ou au manche à balai (joystick). Mais que l'utilisateur qui ne possède pas de joystick se rassure, la plupart les jeux qui seront présentés pourront être manipulés grâce au clavier, c'est-à-dire dans la version de base de votre COMMODORE 64, et dans le cas contraire il vous sera très facile, à partir des autres programmes, de récrire une séquence d'instructions qui adaptera le programme au mode d'utilisation que vous aurez choisi.

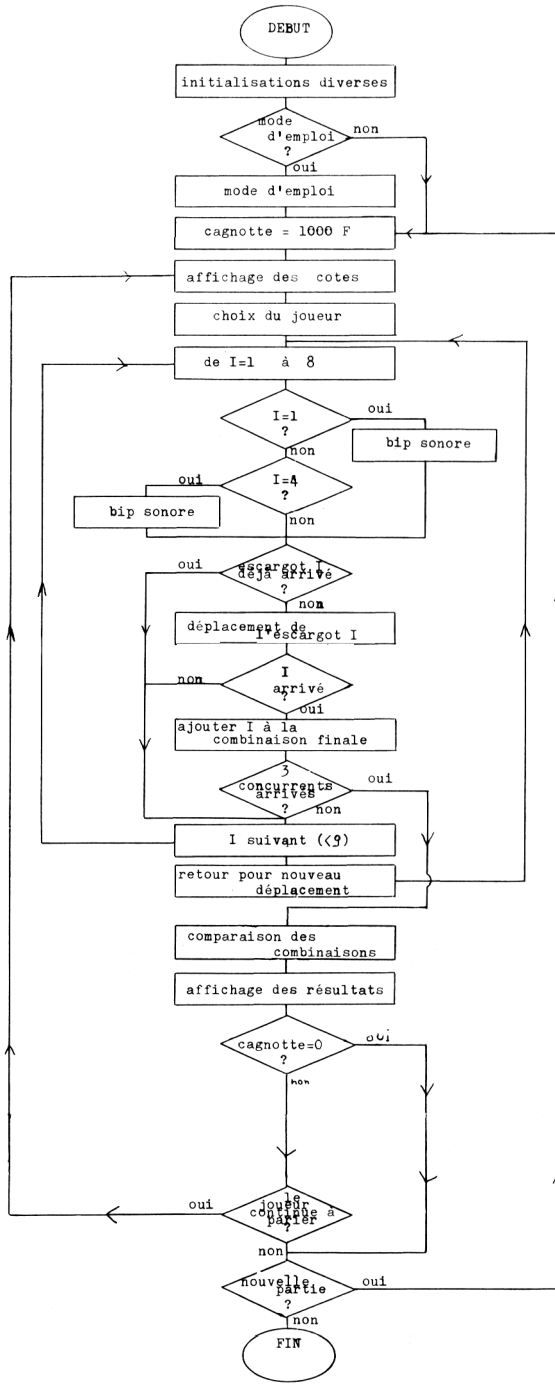
Tous les jeux sont accompagnés des commentaires détaillés de chaque bloc logique, et dans chaque bloc, d'annotations sur pratiquement toutes les lignes (suivant leur importance). Vous disposez également d'un organigramme du programme principal qui vous permet d'avoir une vision synthétique du jeu, ce qui est indispensable à la compréhension.

CHAPITRE 3



3.1 Course d'escargots

Voici un jeu qui va vous rappeler le tiercé tel que vous le connaissez peut-être. Huit escargots sont en ligne pour une course effrénée. Comme pour une véritable course du tiercé, vous disposez des cotes des escargots et vous pariez sur trois des huit gastéropodes. Si vous obtenez l'arrivée dans l'ordre, vous gagnez 100 fois votre mise. Dans le désordre, vous gagnez 10 fois votre mise...


DEBUT
initialisations diverses
mode d'emploi ?
non
oui
mode d'emploi
cagnotte = 1000 F
affichage des cotes
choix du joueur
de I=1 à 8
I=1 ?
oui
bip sonore
non
I=4 ?
oui
bip sonore
non
escargot I déjà arrivé ?
oui
non
déplacement de l'escargot I
I arrivé ?
non
oui
ajouter I à la combinaison finale
3 concurrents arrivés ?
oui
non
I suivant (<9)
retour pour nouveau déplacement
comparaison des combinaisons
affichage des résultats
cagnotte=0 ?
oui
non
le joueur continue à parier ?
oui
non
nouvelle partie ?
oui
non
FIN

Le parieur dispose au départ d'une somme de 1000 francs dont il fait ce qu'il veut: libre à lui de jouer tout, en un seul coup ou de miser franc par franc !

Quand la course est terminée, l'ordinateur affiche les gains éventuels et ce qu'il reste dans votre cagnotte. L'appareil vous demande également si vous désirez continuer à parier.

Lorsque vous n'avez plus d'argent le jeu s'arrête et la machine vous demande si vous désirez faire une autre tentative.

Structure du programme

Instructions préliminaires: lignes 10 à 80

Lignes 10-40: nous fixons ici différents paramètres du programme: couleur du fond, du cadre, effacement des sprites qui se trouveraient éventuellement sur l'écran, effacement de l'écran, initialisation des variables relatives aux sons et aux sprites, et du générateur de nombres aléatoires à l'aide de la fonction temps TI.

Ligne 50: lecture à vide des 30 premières valeurs contenues dans les lignes de DATA.

Ligne 60: appel du sous-programme de mode d'emploi.

Lignes 70-80: initialisation de diverses variables et appel au sous-programme d'affichage

des cotes.

Programme principal : lignes 100 à 230

Ligne 100 : début de la boucle de traitement du déplacement des 8 escargots.

Lignes 110-120 : génération des sons qui vont rythmer l'avancée des escargots. Il y aura deux "bips" par cycle de déplacement : l'un correspondant au premier escargot, et l'autre au quatrième, de façon à équilibrer leur fréquence d'émission.

Ligne 130 : AR est une variable qui contient le nombre d'escargots qui ont franchi la ligne d'arrivée. Si ce nombre est nul, il n'est pas nécessaire d'effectuer le test suivant.

Lignes140-150 : Ce test consiste à aller directement à l'escargot suivant et ainsi éviter de faire avancer un concurrent qui est déjà arrivé ! Si on ne trouve pas d'escargot déjà arrivé, on continue le traitement du déplacement.

Ligne 160 : cette ligne additionne un nombre aléatoire dont l'amplitude dépend de la cote à la coordonnée de l'escargot correspondant.

Ligne 170 : il convient de tester si le concurrent a passé ou non le cap de 255 (l'abscisse de ce dernier). Dans l'affirmative, il faut exécuter l'instruction POKE 53248+16, ES où ES représente les sprites qui ont franchi 255. (pour connaître la valeur de ES se reporter à l'explication détaillée qui figure dans tout manuel d'utilisation).

Ligne 180: cette instruction commande le déplacement horizontal des différents sprites.

Ligne 190: fin de la modulation des "bips" qui scandent la progression des escargots.

Ligne 200: il s'agit de tester si un concurrent est arrivé ou non. Si oui, il faut le placer dans la combinaison gagnante et incrémenter le nombre des escargots déjà arrivés.

Ligne 210: si le nombre de participants déjà arrivés est de trois, la course est finie, et l'on peut alors aller afficher les résultats.

Lignes 220-230: on traite ensuite le déplacement de l'escargot suivant, et lorsque les huit ont avancé, on recommence l'opération en retournant à la ligne 100.

Sous-programme de mode d'emploi: lignes 6000 à 6180

Lignes 6000-6150: libellé du mode d'emploi du jeu.

Lignes 6160-6180: définition des sprites graphiques.

Sous-programme d'initialisation du jeu: lignes 2000 à 5100

Ce sous-programme contient plusieurs phases: l'affichage des cotes dans un premier temps, puis le choix d'une combinaison pour le parieur et la somme qu'il mise, et enfin la mise en place du champ de course et des escargots.

Affichage des cotes: lignes 2000-2060

La cote de chaque concurrent est déterminée aléatoirement entre 3 et 30 contre un. Cette cote influe sur la plage de possibilité de déplacement de l'escargot considéré, ce qui se traduit par CT (I) (que l'on retrouve à la ligne 160 du programme principal). Elle est ensuite imprimée à l'écran.

Choix du joueur: lignes 3000-3060

Le tiercé du parieur est stocké dans la matrice CH (I), et cette partie fait appel au sous-programme qui se trouve aux lignes 7000-7040, et qui teste si vous ne mettez pas le même escargot deux ou trois fois dans la combinaison gagnante...

Mise du joueur: lignes 4000-4030

La mise du parieur est contrôlée de façon à ce que l'on ne dépasse pas le crédit ou que l'on ne mise pas moins de un franc. La mise est ensuite soustraite de la cagnotte.

Apparition des concurrents: lignes 5000-5090

Dans les lignes 5010-540, les lignes de départ et d'arrivée sont imprimées sur l'écran. Les lignes 5050-5090 correspondent à l'apparition des escargots sur la ligne de départ.

Sous-programme d'affichage des résultats: lignes 300 à 530

Lignes 300-340: Déroulement de deux gammes pour marquer la fin du jeu (on lit les valeurs de hautes et basses fréquences dans les lignes de DATA).

Lignes 400-430: effacement de l'écran et disparition de tous les sprites, puis affichage du résultat final.

Lignes 440-490: ces lignes comparent l'arrivée à la combinaison que vous aviez donnée lors des paris. La ligne 490 fait appel à trois sous-programmes possibles suivant que vous avez trouvé l'arrivée dans l'ordre, dans le désordre, ou que vous ne l'avez pas trouvée.

Lignes 500-580: le gain obtenu est affiché et votre cagnotte est mise à jour. Si vous n'avez plus d'argent le jeu s'arrête, mais la machine vous demande tout de même si vous désirez jouer de nouveau (donc avec 10 francs). S'il vous reste de l'argent, la machine vous demande si vous voulez continuer à jouer. Dans l'affirmative, vous gardez la cagnotte que vous aviez acquise.

```
10 PRINT"■":PRINT"■":POKE 53280,0:POKE 53281,0
20 C=53248:POKE C+21,0:T=1:F=0:ZZ=RND(TI)
30 V=54296:M=54276:A=54277:H=54273
40 L=54272:S=54278:POKE V,15:POKE A,9:POKE S,0
50 FOR I=1 TO 30:READ Q:NEXT
60 GOSUB 6000
70 CA=1000
80 GOSUB 2000:K=0:W=1:FOR I=1 TO 8:Z(I)=F:NEXT:AR=0
100 FOR I=1 TO 8
110 IF I=1 THEN POKE H,45:POKE L,198:POKE M,33
120 IF I=4 THEN POKE H,34:POKE L,75:POKE M,33
130 IF AR=0 THEN 160
140 FOR J=1 TO AR:IF I=PH(J) THEN 220
150 NEXT J
160 DX(I)=INT(CT(I)*RND(1)):X(I)=X(I)+DX(I)
170 IF X(I)>255 THEN X(I)=X(I)-256:Z(I)=T:POKE C+16,PEEK(C+16)+2↑(I-1)
180 POKE C+2*I-2,X(I)
190 POKE M,32
200 IF X(I)>57 AND Z(I)=T THEN GOSUB 700
210 IF AR=3 THEN 300
220 NEXT I
230 GOTO 100
300 RESTORE:FOR S=1 TO 15:READ N1,N2
310 POKE H,N1:POKE L,N2:POKE M,33
320 FOR TE=1 TO 20:NEXT TE
330 POKE M,32:FOR TE=1 TO 10:NEXT TE
340 NEXT S
400 PRINT"■":POKE C+21,0:POKE C+16,0:PRINT"VOICI LA COMBINAISON GAGNANTE :"
410 PRINT"■■PREMIER   :",PH(1)
420 PRINT"■SECOND    :",PH(2)
430 PRINT"■TROISIEME :",PH(3)
440 K=0:FOR I=1 TO 3:IF PH(I)=CH(I) THEN K=K+1
450 NEXT:IF K=3 THEN O=1:GOTO 490
460 K=0:FOR I=1 TO 3:FOR J=1 TO 3:IF PH(I)=CH(J) THEN K=K+1
470 NEXT:NEXT:IF K=3 THEN O=2:GOTO 490
480 O=3
490 ON O GOSUB 800,900,1000
500 PRINT"■■VOTRE GAIN EST DE  :";GA;" FRANCS":CA=CA+GA
510 PRINT"■■VOTRE CAGNOTTE SE MONTE A PRESENT A :■":PRINT CA;"FRANCS"
520 PRINT"■■POUR CONTINUER APPUYER SUR UNE TOUCHE"
530 GET R$:IF R$="" THEN 530
540 PRINT"■":IF CA>0 THEN 590
550 PRINT"■■■■■■■■DOMMAGE, VOUS N'AVEZ PLUS D'ARGENT..."
560 PRINT"■■VOULEZ-VOUS JOUER DE NOUVEAU A CE JEU ?":GOSUB 8000
570 IF R$="N" THEN 620
580 GOTO 70
590 PRINT"■■■■■■■■■■VOULEZ-VOUS CONTINUER A PARIER ? (O/N)"
600 GOSUB 8000
610 IF R$="O" THEN 80
620 PRINT"■":END
700 POKE C+21,PEEK(C+21)-2↑(I-1)
710 AR=AR+1:PH(AR)=I:RETURN
800 PRINT"■■VOUS AVEZ TROUVE L'ARRIVEE DANS L'ORDRE."
810 GA=MI*100:RETURN
900 PRINT"■■VOUS AVEZ TROUVE L'ARRIVEE DANS LE■":PRINT"DESORDRE"
910 GA=MI*10:RETURN
1000 PRINT"■■VOUS N'AVEZ PAS TROUVE LA BONNE":PRINT"■COMBINAISON"
1010 GA=0:RETURN
2000 PRINT"■":PRINT"VOICI LES COTES DES CONCURRENTS :■"
2010 FOR I=1 TO 8:RR=3+INT(28*RND(1))
2020 CT(I)=8+INT((30-RR)/5)
2030 PRINT"■NUMERO ";I;" --> ";RR;" CONTRE 1"
```

```
2040 NEXT
2050 PRINT"■■■POUR CONTINUER APPUYER SUR UNE TOUCHE"
2060 GET R$:IF R$="" THEN 2060
3000 PRINT"□":PRINT"QUELLE EST VOTRE COMBINAISON ?"
3010 PRINT"■■■PREMIER   -->";:I=1:GOSUB 7000
3020 PRINT"■■■SECOND    -->";:I=2:GOSUB 7000
3030 PRINT"■■■TROISIEME -->";:I=3:GOSUB 7000
3040 PRINT"■■■■■■■VALIDATION ? (O/N)"
3050 GOSUB 8000
3060 IF R$="N" THEN 3000
4000 PRINT"□":PRINT"■■■■■■■COMBIEN MISEZ-VOUS ?"
4010 PRINT"■(ENTRE 1 ET ";STR$(CA);" FRANCS)"
4020 INPUT MI$:MI=VAL(MI$):IF MI<1 OR MI>CA THEN 4020
4030 CA=CA-MI
5000 PRINT"□":PRINT"■■■■1■■■■2■■■■3■■■■4■■■■5■■■■6■■■■7■■■■8"
5010 FOR I=1029 TO 1989 STEP 40
5020 POKE I,93:POKE I+54272,1:NEXT
5030 FOR I=1063 TO 2023 STEP 40
5040 POKE I,93:POKE I+54272,1:NEXT
5050 POKE C+16,0
5060 FOR I=1 TO 8:Y(I)=25+25*I:X(I)=34
5070 POKE C+2*I-2,34:POKE C+2*I-1,Y(I)
5080 POKE C+38+I,I:NEXT
5090 POKE C+21,255
5100 RETURN
6000 PRINT"□":PRINT"COURSE D'ESCARGOTS"
6010 PRINT"■■■■■■■■■■■VOULEZ-VOUS LES REGLES DU JEU ? (O/N)"
6020 GOSUB 8000
6030 IF R$="N" THEN 6160
6040 PRINT"□":PRINT"IL S'AGIT D'UNE COURSE DE 8 ESCARGOTS"
6050 PRINT "SUR LAQUELLE VOUS MISEZ DE LA MEME FACON"
6060 PRINT"QUE DANS UN TIERCE."
6070 PRINT"■■VOUS DISPOSEZ D'UNE CAGNOTTE DE "
6080 PRINT"1000 FRANCS AU DEPART, ET VOUS MISEZ LA"
6090 PRINT"SOMME QUE VOUS VOULEZ."
6100 PRINT"■■LA COURSE COMMENCE ENSUITE ET VOUS"
6110 PRINT"POUVEZ GAGNER DANS L'ORDRE OU DANS LE"
6120 PRINT"DESORDRE. VOUS MISEZ A L'AIDE DES"
6130 PRINT"COTES QUI VOUS SONT FOURNIES."
6140 PRINT"■■■■■■POUR CONTINUER APPUYER SUR UNE TOUCHE"
6150 GET R$:IF R$="" THEN 6150
6160 FOR N=0 TO 62:READ Q:POKE 832+N,Q:NEXT
6170 FOR I=0 TO 7:POKE 2040+I,13:NEXT
6180 RETURN
7000 GET C$:CH(I)=VAL(C$):IF CH(I)<1 OR CH(I)>8 THEN 7000
7010 IF I=2 AND CH(2)=CH(1) THEN 7000
7020 IF I=3 AND (CH(3)=CH(1) OR CH(3)=CH(2)) THEN 7000
7030 PRINT C$
7040 RETURN
8000 GET R$:IF R$<>"O" AND R$<>"N" THEN 8000
8010 RETURN
9000 DATA 17,37,19,63,21,154,22,227,25,177,28,214,32,94,34,75
9010 DATA 38,126,43,52,45,198,51,97,57,172,64,188,68,149
9020 DATA 0,0,0,0,0,0,0,0,0,0,240,0,3,252,0,7,254,0
9030 DATA 7,14,0,14,247,0,13,247,51,29,151,179,29,91,161,29,219,161
9040 DATA 30,219,161,15,55,18,15,247,12,7,239,31,3,159,255,31,255,255
9050 DATA 127,255,254,255,255,252,0,0,0

READY.
```

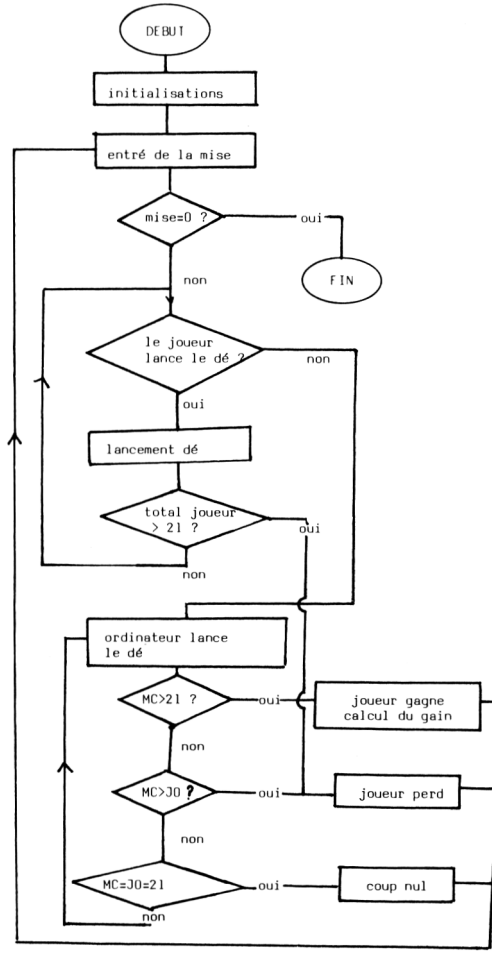
3.2 Jeu du 21

Les règles de ce jeu très connu sont simples. Au départ vous possédez mille francs. Après avoir misé la somme que vous désirez, vous pouvez lancer le dé autant de fois que vous le voulez pour que votre total se raproche le plus possible de 21, sans toutefois dépasser cette valeur.

Pour lancer le dé, pressez simplement la touche RETURN. Lorsque vous jugerez votre total suffisant, entrez une lettre ou un chiffre quelconque avant de presser RETURN. Le COMMODORE lance alors le dé pour son propre compte.

Il détermine ensuite le gagnant et selon le cas augmente ou diminue votre avoir... Une nouvelle partie est ensuite lancée.

Si par malheur votre avoir s'annule, le jeu s'arrête (il vous est, en effet, impossible de miser par manque d'argent). Après la ruine ou l'arrêt du jeu par l'introduction d'une mise nulle, une nouvelle partie vous est proposée: pressez soit la touche O (la lettre) pour rejouer, soit la touche N pour arrêter le jeu définitivement.


DEBUT
initialisations
entré de la mise
mise=0 ?
oui
FIN
non
le joueur lance le dé ?
non
oui
lancement dé
total joueur > 21 ?
oui
non
ordinateur lance le dé
MC>21 ?
oui
joueur gagne calcul du gain
non
MC>JO ?
oui
joueur perd
non
MC=JO=21
oui
coup nul
non

Structure du programme

programme principal: lignes 10 à 530

Ligne 10: appel du sous-programme d'initialisation.

Ligne 20: le joueur se voit crédité d'un avoir de 10 francs.

Ligne 30: boucle de temporisation.

Ligne 40: effacement de l'écran, initialisation de variables.

Ligne 50: affiche l'avoir du joueur.

Ligne 60: demande la mise.

Ligne 70: protection pour éviter que la mise ne soit décimale ou négative.

Ligne 80: toute mise supérieure à l'avoir du joueur est refusée.

Ligne 90: une mise nulle renvoie à la fin de la partie.

Ligne 100: le joueur veut-il lancer le dé?

Ligne 110: s'il répond par un caractère l'ordinateur va alors lancer le dé pour son propre compte.

Ligne 120: lance le dé pour le joueur, appelle le sous-programme de temporisation.

Ligne 130: affichage du total du joueur.

Ligne 140: si le total du joueur dépase 21, il a perdu.

Ligne 150: retour à la ligne 100 pour éventuellement un nouveau lancer.

Ligne 160: le Commodore 64 lance le dé pour son propre compte.

Ligne 170: appel du sous-programme de temporisation.

Ligne 180: affichage du total du Commodore 64.

Ligne 200: si le total du Commodore dépasse 21, le joueur gagne.

Ligne 210: si le total du Commodore dépasse le total du joueur, ce dernier perd.

Ligne 220: si le joueur et le Commodore ont obtenu tous les deux 21, il y a coup nul.

Ligne 230: temporisation et retour au début de la boucle de jeu.

Ligne 300: affichage du message de gain.

Ligne 310: l'avoir est augmenté de une à cinq fois la mise.

Ligne 320: retour en 30 pour une nouvelle partie.

Ligne 400: affichage du message de perte.

Ligne 410: diminution de la mise.

Ligne 420: si le joueur n'a pas un avoir nul, retour en 30 pour une nouvelle partie.

Ligne 430: affichage du message de ruine.

Lignes 500-530: l'ordinateur demande si le joueur désire faire une autre partie.

Sous-programme de temporisation: ligne 1000

Il est composé d'une boucle de temporisation.

Sous-programme d'initialisation: lignes 2000 à 2030

Ligne 2000: efface l'écran et affiche le titre du jeu.

Lignes 2010-2020: initialisation du générateur de nombre pseudo-aléatoire par un nombre aléatoire (le temps).

Ligne 2030: fin du sous-programme.

```
10 GOSUB 2000
20 AV=1000
30 FOR I= 1 TO 400 :NEXTI
40 PRINT"□":MC=0:JO=0:K=0
50 PRINT"AVOIR : ";AV:PRINT
60 PRINT:INPUT"MISE ";MI
70 MI=INT(ABS(MI))
80 IF MI>AV THEN 30
90 IF MI= 0 THEN 500
100 J$="":PRINT:INPUT"JET ";J$
110 IF J$<>"" THEN 160
120 JO = JO+INT(RND(1)*6+1):GOSUB 1000
130 PRINT"JOUEUR : ";JO
140 IF JO >21 THEN 400
150 GOTO100
160 MC=MC+INT(RND(1)*6+1)
170 GOSUB1000
180 PRINT"MICRO : ";MC
190 K=K+2:IF K=24 THEN K=0
200 IF MC>21 THEN 300
210 IF MC>JO THEN 400
220 IF MC+JO-42=0 THEN PRINT"COUP NUL !!":FOR I=1 TO 1000:NEXT:GOTO30
230 FOR I= 1 TO 100:NEXT:GOTO160
300 PRINT"VOUS GAGNEZ !!":FOR I= 1 TO 1000:NEXT
310 AV=AV+INT(RND(1)*5+1)*MI
320 GOTO30
400 PRINT"VOUS PERDEZ ...":FOR I=1 TO 1000:NEXT
410 AV=AV-MI
420 IF AV>0 THEN 30
430 PRINT"VOUS ETES RUINE !!!"
500 PRINT "VOULEZ-VOUS REJOUER ?"
510 GETR$:IF R$<>"O" AND R$<>"N" THEN 510
520 IF R$="O" THEN 20
530 PRINT"□":END
1000 FOR I=1 TO 500:NEXT:RETURN
2000 PRINT"□":PRINT"■■■■■■■■■■■■■■■■■■■■JEU DU 21■■"
2010 PRINT"■■■■■■■■■■■■ENTREZ UN NOMBRE ■":INPUT A
2020 A=RND(TI)
2030 RETURN

READY.
```

3.3 Balle au pied

Ce jeu, de pur hasard, est de manipulation très simple. Un robot lance au hasard une balle qui tombe dans une des cases dessinées en bas de l'écran. Le joueur doit parier sur une des couleurs. Si la balle tombe dans une des cases de la couleur choisie, le gain est égal à la mise multipliée par le chiffre inscrit dans la case.

Vous remarquerez que les gains ne sont pas tous égaux mais qu'ils sont plus ou moins inversement proportionnels à une certaine probabilité de tomber dans la case choisie.

Messieurs, faites vos jeux.

Stucture du programme

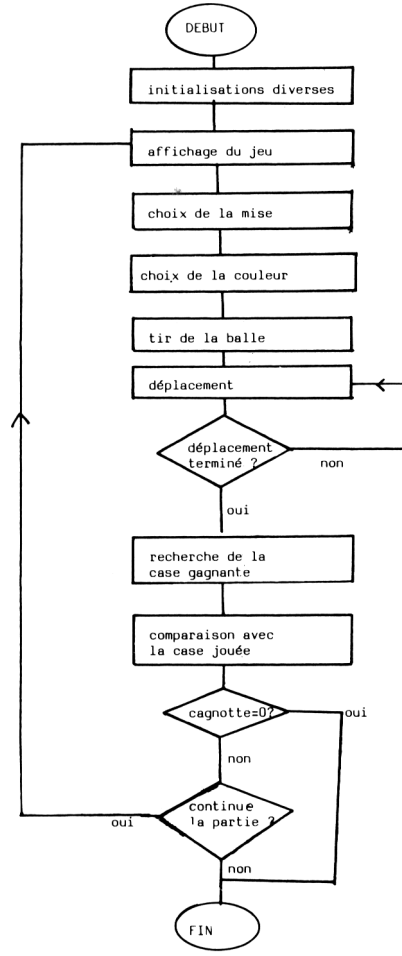
Programme principal: lignes 100 à 660

Lignes 100: initialisation de la case de départ.

Ligne 110: appel du sous-programme de mise en page.

Ligne 120: appel du sous-programme de jeu.

Ligne 130: apparition du tir du robot.


DEBUT
initialisations diverses
affichage du jeu
choix de la mise
choix de la couleur
tir de la balle
déplacement
déplacement terminé ?
non
oui
recherche de la case gagnante
comparaison avec la case jouée
cagnotte=0?
oui
non
continue la partie ?
oui
non
FIN

Lignes 140-150: initialisation des coordonnées de la balle.

Lignes 160-170: apparition de la balle.

Lignes 180-190: choix aléatoire du déplacement de la balle.

Lignes 200-210: mise à jour des coordonnées de la balle.

Ligne 220: si la balle est en bas, fin de la boucle de déplacement de celle-ci.

Ligne 230: modification du paramètre de déplacement vertical pour donner un effet de chute libre.

Lignes 240-250: déplacement de la balle.

Ligne 260: retour au début de la boucle de chute de la balle.

Ligne 270: remise du robot à l'état initial.

Lignes 280-370: recherche de la case où est tombée la balle.

Ligne 380: boucle d'attente.

Lignes 390-420: affichage du résultat.

Ligne 430: si le gain est nul, on n'exécute pas la partie de sonorisation.

Lignes 440-500: sonorisation.

Ligne 510: affiche le gain.

Ligne 520: calcule l'argent restant et remet le gain à 0.

Ligne 530: affiche la somme restante.

Ligne 540: si elle est nulle, fin du programme.

Lignes 550-580: fin de partie classique.

Lignes 590-660: sous-programmes de calcul du gain.

Sous-programme de jeu: lignes 1000 à 1110

Lignes 1000-1070: lecture au clavier de la mise et de la couleur choisie.

Lignes 1080-1100: sonorisation de la frappe de la balle par le robot.

Ligne 1110: fin du sous-programme.

Sous-programme de mise en page: lignes 2000 à 2320

Lignes 2000-2190: initialisations diverses, en particulier des sprites.

Ligne 2200: efface l'écran.

Ligne 2210: fait apparaître le robot en position de repos.

Lignes 2220-2310: affichent le terrain de jeu.

Ligne 2320: fin du sous-programme.

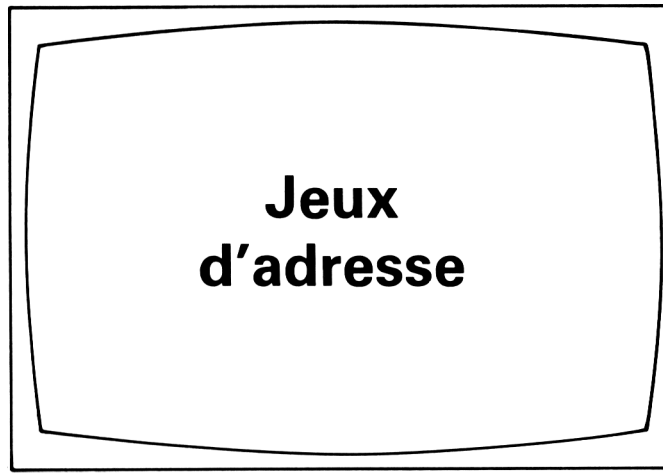
Données relatives à la sonorisation: lignes 10000 & 10010

Données relatives aux sprites: lignes 10010 à 10220

```
0 REM BALLE AU PIED
100 K=0
110 GOSUB 2000
120 GOSUB 1000
130 POKEV+21,6
140 X=80
150 Y=140
160 POKEV+4,X
170 POKEV+5,Y
180 DY=INT(RND(TI)*9)-4:IF DY=0 THEN 180
190 DX=INT(RND(TI)*4)+1
200 X=X+DX:Y=Y+DY
210 IF X>255 THEN DX=-DX:X=X+DX
220 IF Y>225 THEN 270
230 DY=DY+0.1
240 POKEV+5,Y
250 POKEV+4,X
260 GOTO200
270 POKEV+21,5
280 IF X<60 THEN K=0 :GOTO380
290 IF X<84 THEN K=1 :GOTO380
300 IF X<108THEN K=2 :GOTO380
310 IF X<132THEN K=3 :GOTO380
320 IF X<156THEN K=4 :GOTO380
330 IF X<180THEN K=5 :GOTO380
340 IF X<204THEN K=6 :GOTO380
350 IF X<228THEN K=7 :GOTO380
360 IF X<252THEN K=8 :GOTO380
370 K=0
380 FOR I=1 TO 1000:NEXT
390 PRINT"■"
400 POKEV+21,0
410 IF K=0 THEN PRINT"■■■■■ DOMMAGE VOUS ETES TOMBES EN DEHORS DE LA CIBLE":G=0
420 IFK=2*C OR K=C*2-1THEN ON K GOSUB 590,600,610,620,630,640,650,660
430 IF G=0 THEN520
440 POKE54277,65:POKE54278,31
450 RESTORE:FOR S=1 TO 15:READ N1,N2
460 POKE54296,15
470 POKE 54273,N1:POKE 54272,N2:POKE 54276,33
480 FOR TE=1 TO 20:NEXT TE
490 POKE 54276,32:FOR TE=1 TO 10:NEXT TE
500 NEXT S:POKE54296,0
510 PRINT"■■■VOTRE GAIN EST DE :";G
520 RR=RR-R+G:G=0
530 PRINT"■■■IL VOUS RESTE :";RR;"FRANCS"
540 IF RR=0 THEN END
550 PRINT"■■■VOULEZ-VOUS CONTINUER ?"
560 GETR$:IF R$<>"O" AND R$<>"N" THEN GOTO560
570 IF R$="O"THEN GOSUB2060:GOTO120
580 PRINT"■":END
590 G=9*R:RETURN
600 G=8*R:RETURN
610 G=4*R:RETURN
620 G=2*R:RETURN
630 G=2*R:RETURN
640 G=5*R:RETURN
650 G=8*R:RETURN
660 G=9*R:RETURN
1000 INPUT"■MISE ";R$
1010 R=VAL(R$):IF R<1 OR R>RR THEN PRINT"■                    ":GOTO1000
1020 PRINT
1030 PRINT"COULEUR ?";
1040 GETR$:IF R$=""THEN 1040
1050 C=VAL(R$)
1060 IF C<1 OR C>4 THEN GOTO1040
1070 PRINTC
1080 POKE54296,15
```

```
1090 POKE 54273,36:POKE 54272,85:POKE54277,128:POKE54278,8:POKE 54276,129
1100 POKE54276,128
1110 RETURN
2000 RR=1000
2010 V=53248
2020 FOR S=1 TO 15:READ N1,N2:NEXT
2030 FORI=0 TO 62 :READQ:POKE832+I,Q:NEXT
2040 FORI=0 TO 62 :READQ:POKE896+I,Q:NEXT
2050 FORI=0 TO 62 :READQ:POKE960+I,Q:NEXT
2060 POKEV,60
2070 POKEV+2,60
2080 POKEV+4,75
2090 POKEV+1,110
2100 POKEV+3,100
2110 POKEV+5,140
2120 POKE2040,13
2130 POKE2042,15
2140 POKE2041,14
2150 POKEV+39,2
2160 POKEV+40,2
2170 POKEV+41,7
2180 POKEV+29,3
2190 POKEV+23,3
2200 PRINT"■"
2210 POKEV+21,5
2220 FOR I=1 TO 19:PRINTTAB(30)"■■ ■■":NEXT
2230 PRINT
2240 PRINT"       ■■ 9 ▌8 ■ 4 ▌2 ■ 2 ▌4 ■ 8 ▌9 ■■"
2250 PRINT"       ■■_▌_■_▌_■_▌_■_▌_■■"
2260 PRINT"■■COULEUR■■1  1   2   2   3   3   4   4"
2270 PRINT"■■■■":PRINTTAB(34)"■■PARI■■"
2280 PRINT
2290 PRINT
2300 Z=Z+1
2310 PRINTTAB(35)Z
2320 RETURN
10000 DATA 17,37,19,63,21,154,22,227,25,177,28,214,32,94,34,75
10010 DATA 38,126,43,52,45,198,51,97,57,172,64,188,68,149
10020 DATA 60,0,0,52,0,0,62,0,0
10030 DATA 60,0,0,60,0,0,24,0,0
10040 DATA 126,0,0,126,0,0,126,0,0
10050 DATA 126,0,0,126,0,0,126,0,0
10060 DATA 126,0,0,60,0,0,28,0,0
10070 DATA 28,0,0,28,0,0,28,0,0
10080 DATA 28,0,0,28,0,0,31,0,0
10090 DATA 60,0,0,52,0,0,62,0,0
10100 DATA 60,0,0,60,0,0,24,0,0
10110 DATA 126,0,0,126,0,0,126,0,0
10120 DATA 126,0,0,126,0,0,126,0,0
10130 DATA 126,0,0,60,0,0,30,0,0
10140 DATA 31,16,0,27,184,0,25,240,0
10150 DATA 24,224,0,24,64,0,31,0,0
10160 DATA 14,0,0,63,128,0,63,128,0
10170 DATA 127,192,0,255,224,0,255,224,0
10180 DATA 255,224,0,127,192,0,63,128,0
10190 DATA 63,128,0,14,0,0,0,0,0
10200 DATA 0,0,0,0,0,0,0,0,0
10210 DATA 0,0,0,0,0,0,0,0,0
10220 DATA 0,0,0,0,0,0,0,0,0

READY.
```

CHAPITRE 4



4.1 Squash

Vous connaissez déjà ce jeu qui vous a été présenté durant l'exposé de la première partie. Nous ne fournirons pas ici l'organigramme de ce jeu puisqu'il se trouve déjà dans le début de cet ouvrage.

La raquette peut être déplacée soit à l'aide du joystick, soit par l'intermédiaire du clavier. Dans le second cas, les touches à utiliser sont les suivantes:

* vers le bas : touche M,

* pour arrêter la raquette : touche N,

* vers le haut : touche B.

Structure du programme

Intructions préliminaires: lignes 10 à 30

Ligne 10: appel du sous-programme d'initialisation des sprites.

Ligne 20: appel du sous-programme de mode d'emploi.

Ligne 30: initialisation des variables associées à la grande raquette.

Lignes 100-140: émission d'un son lors de la perte d'une balle. La première fois que l'on joue, il ne faut pas exécuter cette séquence d'instructions, d'où le test de la ligne 100.

Ligne 150: Si la balle perdue était la troisième, alors le jeu est terminé, il faut aller en 400, où sera affiché le score.

Lignes 160-180: il s'agit de l'initialisation des sprites qui sont la balle et les raquettes, et des paramètres qui leur sont associés.

Programme principal: lignes 200 à 310

Ligne 200: appel du sous-programme de déplacement suivant le mode choisi.

Ligne 210: progression des valeurs des coordonnées des sprites.

Ligne 220: le déplacement de la raquette est limité par les valeurs extrêmes indiquées. Si la valeur n'est pas dans l'intervalle voulu, on reprend la valeur précédente, et on ne passe pas sur l'instruction d'affichage du sprite.

Ligne 230: affichage du sprite-raquette, à sa nouvelle place.

Lignes 240-260: affichage de la balle à son nouvel emplacement. Le test prend en compte le fait que lorsque l'abscisse du sprite est supérieure à 255, il faut repartir à 0 en exécutant l'instruction POKE 53248+16, N, où N est le numéro en base 2 du sprite correspondant.

Ligne 270: test de rebond de la balle sur les bords horizontaux.

Ligne 280: si la balle passe derrière la raquette, le joueur perd une balle, et l'on retourne au début du jeu.

Ligne 290: test de rebond de la balle sur le mur du fond.

Ligne 300: la machine teste si la balle se trouve à la hauteur de la raquette. Dans l'affirmative, elle exécute le sous-programme situé à la ligne 2500.

Ligne 310: fin de la boucle du programme.

Affichage des résultats: lignes 400 à 460

Ligne 400: effacement de l'écran et des sprites, changement de couleur du fond et du cadre.

Lignes 410-460: donnée du score final et demande de la machine pour une nouvelle partie éventuelle.

Utilisation du joystick: lignes 1000 à 1070

Ligne 1000: la configuration du joystick est saisie dans la mémoire qui porte le numéro 56321, sur le port 2.

Ligne 1030: sélection de la direction à prendre suivant la valeur de D obtenue.

Lignes 1040-1070: affectation de la valeur de déplacement choisie.

Utilisation du clavier: lignes 2000 à 2030

Ligne 2000: saisie de la lettre tapée au clavier.

Lignes 2000-2020: affectation de la valeur de déplacement suivant la lettre.

Rebond de la balle sur la raquette: lignes 2500 à 2700

Ligne 2500: si la balle ne touche pas la raquette alors il faut sortir du sous-programme.

Ligne 2510: appel du sous-programme de sonorisation, qui marque le rebond de la balle sur la raquette.

Ligne 2520: si le rebond a lieu au milieu de la raquette, la balle repart normalement (même angle).

Ligne 2530: si la balle arrive sur un des bords de la raquette, la balle repart avec un angle aléatoire.

Ligne 2540: la gestion du score s'effectue par incrémentation de K, et lorsque cette variable atteint une valeur multiple de 20, la raquette change de dimension, par appel de l'un des sous-programmes situés en 2600 et 2700.

Ligne 2600: sous-programme de changement de raquette, la raquette devient grande.

Ligne 2700: sous-programme de changement de raquette, la raquette devient petite.

Sous-programme générant des angles aléatoires: lignes 3000 à 3060

Ligne 3000: choix aléatoire de l'une des trois possibilités de valeur pour D (variable de transition qui sera affectée à la valeur du déplacement vertical de la balle).

Lignes 3010-3060: détermination de la valeur de D, et de son signe (ce dernier étant aléatoire).

Génération de sons: lignes 3500 à 3530

Ce sous-programme est appelé chaque fois que la balle rebondit sur un mur ou sur la raquette. On fixe ici les valeurs des hautes et basses fréquences et la modulation du son à émettre.

Choix du mode de déplacement: lignes 4000 à 4080

Ce sous-programme qui est exécuté au début du jeu, vous permet de choisir si vous allez mouvoir votre raquette à l'aide du clavier ou avec le joystick. Il permet aussi d'initialiser les variables associées aux sons et au graphisme. ainsi que le générateur de nombres pseudo-aléatoires.

Initialisation des sprites: lignes 5000 à 5040

La formation des trois sprites, correspondant à la balle, la petite raquette et la grande, fait appel aux lignes de DATA de la fin du programme. On détermine ligne 5030 les couleurs des trois sprites.

```
10 GOSUB 5000
20 GOSUB 4000
30 DI=2:POKE C+21,DI+1:L1=21:L2=3:L3=10
100 IF NB=0 THEN 160
110 POKE HH,19:POKE LL,63
120 POKE MM,17
130 FOR TE=1 TO 2000:NEXT
140 POKE MM,16
150 IF NB=3 THEN 400
160 X=328:Y=58+INT(180*RND(1)):GOSUB 3000:DY=D:XR=32:YR=144:DR=0
170 POKE C+21,DI+1:POKE C+16,1:POKE C,X-256:POKE C+1,Y
180 POKE C+DI,XR:POKE C+DI+1,YR
200 ON M GOSUB 1000,2000
210 X=X+DX:Y=Y+DY:YR=YR+DR
220 IF YR<25 OR YR>254 THEN YR=YR-DR:GOTO 240
230 POKE C+DI+1,YR
240 IF X<256 THEN POKE C+16,0:POKE C,X:GOTO 260
250 POKE C+16,1:POKE C,X-256
260 POKE C+1,Y
270 IF (Y<51 AND Y>49+DY) OR (Y>241 AND Y<243+DY) THEN GOSUB 3500:DY=-DY
280 IF X<21 THEN NB=NB+1:GOTO 30
290 IF X>327 THEN GOSUB 3500:DX=-DX
300 IF X=40 THEN GOSUB 2500
310 GOTO 200
400 PRINT"□":POKE 53280,0:POKE 53281,0:POKE C+21,0
410 PRINT"VOUS AVEZ OBTENU UN SCORE DE : "
420 FOR I=1 TO 5:PRINT:NEXT:PRINT K,"POINTS"
430 FOR I=1 TO 5:PRINT:NEXT:PRINT"VOULEZ-VOUS REJOUER ? (O/N)"
440 GET R$:IF R$<>"O" AND R$<>"N" THEN 440
450 IF R$="O" THEN 20
460 PRINT"□":END
1000 D=PEEK(56321):D=255-D:IF D=0 THEN DR=0:RETURN
1030 ON D GOSUB 1070,1060,1050,1050,1050,1050,1050,1050,1050,1050
1040 RETURN
1050 DR=0:RETURN
1060 DR=8:RETURN
1070 DR=-8:RETURN
2000 GET R$:IF R$="M" THEN DR=8
2010 IF R$="N" THEN DR=0
2020 IF R$="B" THEN DR=-8
2030 RETURN
2500 IF Y+7<YR OR Y>YR+L1 THEN RETURN
2510 GOSUB 3500
2520 IF Y>YR+L2 AND Y<YR+L3 THEN DX=-DX:GOTO 2540
2530 GOSUB 3000:DX=-DX:DY=D
2540 K=K+1:IF K=20*H THEN ON 1+H-2*INT(H/2) GOSUB 2600,2700:H=H+1
2550 RETURN
2600 DI=2:L1=20:L2=3:L3=10:POKE C+21,DI+1:POKE C+DI,XR:POKE C+DI+1,YR:RETURN
2700 DI=4:L1=7:L2=-5:L3=5:POKE C+21,DI+1:POKE C+DI,XR:POKE C+DI+1,YR:RETURN
3000 ON INT(1+3*RND(1)) GOTO 3010,3030,3050
3010 D=8:IF RND(1)<0.5 THEN D=-D
3020 RETURN
3030 D=4:IF RND(1)<0.5 THEN D=-D
3040 RETURN
3050 D=2:IF RND(1)<0.5 THEN D=-D
3060 RETURN
3500 POKE HH,45:POKE LL,198
3510 POKE MM,17
3520 FOR TE=1 TO 10:NEXT:POKE MM,16
3530 RETURN
4000 PRINT"□":POKE 53280,0:POKE 53281,0:C=53248:POKE C+21,0:ZZ=RND(TI)
4010 VV=54296:MM=54276:AA=54277:HH=54273:LL=54272:SS=54278:POKE VV,15
```

```
4020 POKE AA,0:POKE SS,240
4050 PRINT"JEU DE SQUASH"
4060 PRINT"■■■MODE DE DEPLACEMENT :"
4070 PRINT"■■MANCHE A BALAI : 1"
4080 PRINT"■CLAVIER          : 2"
4090 GET M$:M=VAL(M$):IF M<>1 AND M<>2 THEN 4090
4100 PRINT"■■■VITESSE DE LA BALLE :"
4110 PRINT"■■LENT     : 1"
4120 PRINT"■RAPIDE    : 2"
4130 GET VI$:VI=VAL(VI$):IF VI<>1 AND VI<>2 THEN 4130
4140 DX=-8*VI
4150 PRINT"■■■POUR COMMENCER APPUYER SUR UNE TOUCHE"
4160 GET R$:IF R$="" THEN 4160
4170 PRINT"■":POKE 53280,6:POKE 53281,7:POKE C+21,0:NB=0:K=0:H=1
4180 FOR I=1063 TO 2023 STEP 40:POKE I,160:POKE 54272+I,6:NEXT
4190 RETURN
4200 RETURN
5000 POKE 2040,13:FOR N=0 TO 62:READ Q:POKE 832+N,Q:NEXT
5010 POKE 2041,14:FOR N=0 TO 62:READ Q:POKE 896+N,Q:NEXT
5020 POKE 2042,15:FOR N=0 TO 62:READ Q:POKE 960+N,Q:NEXT
5030 POKE C+40,4:POKE C+41,4:POKE C+39,0
5040 RETURN
6000 DATA 60,0,0,126,0,0,255,0,0,255,0,0,255,0,0,255,0,0
6010 DATA 126,0,0,60,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
6020 DATA 0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
6030 DATA 255,0,0,255,0,0,255,0,0,255,0,0,255,0,0,255,0,0
6040 DATA 255,0,0,255,0,0,255,0,0,255,0,0,255,0,0,255,0,0
6050 DATA 255,0,0,255,0,0,255,0,0,255,0,0,255,0,0,255,0,0
6060 DATA 255,0,0,255,0,0,255,0,0
6070 DATA 255,0,0,255,0,0,255,0,0,255,0,0,255,0,0,255,0,0
6080 DATA 255,0,0,255,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
6090 DATA 0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0

READY.
```

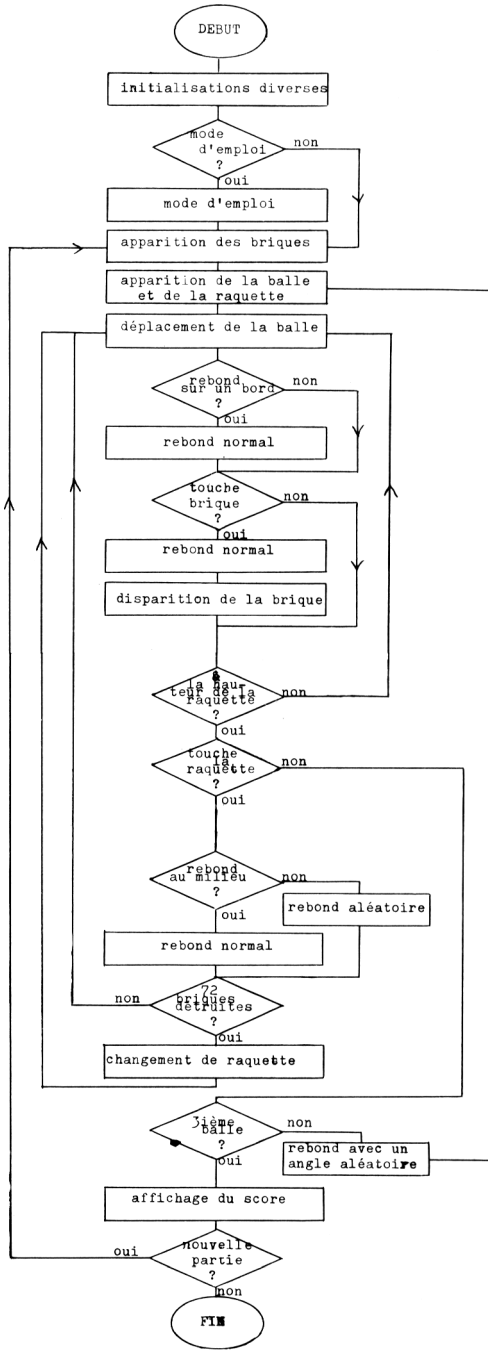
4.2 Casse-briques

Le casse-briques est un jeu que vous connaissez certainement, pour l'avoir vu relativement souvent comme jeu vidéo. Voici une version pour votre micro-ordinateur qui va vous montrer comment on programme ce genre de jeu en BASIC.

Vous disposez donc d'une raquette, de grandeur variable, avec laquelle vous devez envoyer une balle sur deux murs de briques successifs, afin de les détruire. Chaque rebond de la balle sur un mur, à l'aller comme au retour, provoque la destruction de la brique correspondante, et donc son effacement. Vous diposez de trois balles en tout.

Vous pouvez constater que, pour tout ce qui concerne le déplacement de la balle et de la raquette, ce jeu ressemble tout à fait au programme de squash qui a été présenté au dessus. C'est pourquoi nous essaierons, dans la mesure du possible, de conserver les instructions communes de façon à en faciliter la compréhension. Comme le jeu de squash, ce jeu est utilisable soit avec le joystick, soit à l'aide du clavier. Pour l'utilisation au clavier vous allez utiliser les touches :

* vers le bas : touche M,
* pour arrêter la raquette : touche N,
* vers le haut : touche B.


DEBUT
initialisations diverses
mode d'emploi ?
non
oui
mode d'emploi
apparition des briques
apparition de la balle et de la raquette
déplacement de la balle
rebond sur un bord ?
non
oui
rebond normal
touche brique ?
non
oui
rebond normal
disparition de la brique
la balle est à la hauteur de la raquette ?
non
oui
touche la raquette ?
non
oui
rebond au milieu ?
non
rebond aléatoire
oui
rebond normal
72 briques détruites ?
non
oui
changement de raquette
3ième balle ?
non
rebond avec un angle aléatoire
oui
affichage du score
nouvelle partie ?
oui
non
FIN

Structure du programme

Instructions préliminaires: lignes 10 à 100

Ligne 10: appel au sous-programme de mode d'emploi, qui gère également diverses initialisations de variables.

Ligne 20: si NB, c'est-à-dire le nombre de balles déjà utilisées, est nul (début du jeu), alors il est inutile de faire exécuter la séquence d'instructions qui correspond à un effet sonore pour la perte d'une balle.

Lignes 30-50: génération d'un effet sonore dans le cas de la perte d'une balle (on reconnaît ici les contrôles de modulation, de haute et basse fréquence).

Ligne 60: lorsque le nombre de balles alloué a été atteint, il faut aller au bloc qui se charge de l'affichage des résultats. DR est la variable qui contient la valeur du déplacement de la raquette.

Lignes 70-100: initialisation des variables relatives aux sprites (coordonnées, pas des déplacements...) La ligne 80 contient une boucle de temporisation pour éviter que la balle ne vous surprenne au début du jeu.

Programme principal: lignes 200 à 360

Ligne 200: appel des sous-programmes de déplacement de la raquette suivant le mode choisi: clavier ou joystick.

Ligne 230: mise à jour des nouvelles valeurs des coordonnées des sprites. La fin de cette ligne est constituée d'un test qui limite le déplacement de la raquette en haut et en bas de l'écran.

Ligne 240: déplacement de la raquette.

Lignes 250-270: déplacement de la balle.

Ligne 280: la balle passe derrière la raquette, le joueur la perd, et la machine réduit son crédit de une balle.

Lignes 290-300: ce sont les tests qui assurent les rebonds de la balle sur les bords. L'appel au sous-programme situé à la ligne 1000 correspond à l'émission d'un "bip" lors du choc.

Ligne 310: si la balle se trouve à la hauteur de la raquette, il faut exécuter le sous-programme qui se charge des rebonds.

Lignes 320-340: ces trois lignes servent à savoir si la balle se trouve en face d'une brique ou non, afin de la détruire. Si c'est le cas, la disparition de la brique sera assurée par le sous-programme situé en 700.

Ligne 350: si le nombre de briques abattues est de 72, c'est-à-dire le nombre de briques des deux murs, il faut faire apparaître d'autres murs et réinitialiliser certaines variables.

Ligne 360: retour au début de la boucle de traitement.

Ligne 400: réinitialisation de l'écran quand toutes les briques ont été détruites.

Gestion des rebonds sur la raquette: lignes 500 à 530

Ligne 500: si la balle arrive en dehors des limites de la raquette, il faut sortir du sous-programme.

Ligne 510: en cas de rebond, on appelle alors le sous-programme de sons qui émet un "bip".

Ligne 520: si la balle tombe sur le milieu de la raquette, le rebond est normal.

Ligne 530: sinon, la balle repart avec un angle aléatoire, c'est-à-dire un DY aléatoire.

Effacement des briques détruites: lignes 700 à 730

Ligne 700: cette ligne se charge du rebond de la balle, de l'incrémentation du score, et détermine la brique à effacer.

Ligne 710: émission d'un "bip" sonore et choix de la brique à effacer suivant la valeur de P.

Lignes 720-730: effacement de la brique.

Gestion de la taille de la raquette: lignes 800 à 820

Ligne 800: détermine la nouvelle raquette à faire apparaître sur l'écran.

Lignes 810-820: initialisation des différents paramètres suivant la grandeur de la raquette.

Rebonds aléatoires: lignes 900 à 960

Ligne 900: cette ligne détermine aléatoirement la valeur de la variable D.

Lignes 910-960: affectation de la valeur de D, et choix aléatoire de son signe.

Déplacement de la raquette au clavier: lignes 1300 à 1330

Ligne 1300: saisie de la valeur de R$ au clavier pour déterminer la direction de déplacement.

Lignes 1300-1330: choix de la direction suivant R$.

Le joystick: lignes 1500 à 1530:

Ligne 1500: saisie de la valeur de la case mémoire portant le numéro 56321 (port 2 du joystick).

Lignes 1510-1530: valeur du déplacement suivant la valeur de D.

Mode d'emploi: lignes 2000 à 2300

Ce sous-programme contient d'une part un certain nombre d'initialisations de variables (sonores, graphiques...), et d'autre part un mode d'emploi.

Choix du mode de déplacement: lignes 2400 à 2430

Le joueur fait le choix d'un mode de déplacement de la raquette: clavier ou joystick.

Construction de l'écran: lignes 2500 à 2700

Ligne 2500: couleurs du fond et du cadre, effacement des sprites qui se trouveraient éventuellement sur l'écran, effacement de l'écran.

Lignes 2510: raccourcissement de la fenêtre de jeu.

Ligne 2520-2640: affichage à l'écran des murs, en deux rangées successives de trois briques d'épaisseur.

Lignes 2650-2690: diminution de la fenêtre en longueur.

Affichage du score final: lignes 3000 à 3110:

Lignes 3000-3070: affichage des résultats.

Lignes 3080-3110: la machine demande si l'utilisateur veut jouer de nouveau.

Initialisation des sprites: lignes 4000 à 5100

Lignes 4000-4040: lecture des lignes de DATA pour initialiser les sprites graphiques et détermination des couleurs de ces sprites.

Lignes 5000-5100: lignes de données relatives aux sprites.

```
10 GOSUB 2000
20 IF NB=0 THEN 60
30 POKE HH,19:POKE LL,63:POKE MM,17
40 FOR TE=1 TO 2000:NEXT
50 POKE MM,16
60 DR=0:IF NB=3 THEN 3000
70 XR=56:YR=144:POKE C+DI,XR:POKE C+DI+1,YR
80 FOR I=1 TO 1000:NEXT:XB=208:YB=154
90 DX=-8:GOSUB 900:DY=D
100 POKE C+21,8+DI:POKE C+6,XB:POKE C+7,YB
200 ON MO GOSUB 1300,1500
230 XB=XB+DX:YB=YB+DY:YR=YR+DR:IF YR<35 OR YR>254 THEN YR=YR-DR:GOTO 250
240 POKE C+DI+1,YR
250 IF XB<=255 THEN POKE C+16,0:POKE C+6,XB:GOTO 270
260 POKE C+16,8:POKE C+6,XB-256
270 POKE C+7,YB
280 IF XB<50 THEN NB=NB+1:GOTO 30
290 IF XB>335 THEN GOSUB 1000:DX=-DX
300 IF (YB<59 AND YB>57+DY) OR (YB>241 AND YB<243+DY) THEN GOSUB 1000:DY=-DY
310 IF XB=64 THEN GOSUB 500:GOTO 200
320 N=INT(0.5+(XB-24)/8)+DX/8:IF N<23 THEN 200
330 M=INT(0.5+(YB-50)/8)
340 IF PEEK(1024+N+40*M)=160 THEN GOSUB 700
350 IF BB AND K=72*INT(K/72) THEN 400
360 GOTO 200
400 GOSUB 800:GOSUB 2500:GOTO 60
500 IF YB+7<YR OR YB>YR+L1 THEN RETURN
510 GOSUB 1000
520 IF YB>YR+L2 AND YB<YR+L3 THEN DX=-DX:RETURN
530 GOSUB 900:DX=-DX:DY=D:RETURN
700 P=M-2*INT(M/2):E=N+40*M:DX=-DX:K=K+1
710 BB=T:GOSUB 1000
720 IF P=1 THEN POKE 1024+E,32:POKE 1064+E,32
730 IF P=0 THEN POKE 1024+E,32:POKE 984+E,32
740 RETURN
800 KK=INT(K/72):BB=F:ON 1+KK-2*INT(KK/2) GOTO 810,820
810 DI=2:L1=20:L2=3:L3=10:POKE C+21,DI+8:RETURN
820 DI=4:L1=7:L2=-5:L3=5:POKE C+21,DI+8:RETURN
900 D=1+INT(3*RND(1)):ON D GOTO 910,930,950
910 D=8:IF RND(1)<0.5 THEN D=-D
920 RETURN
930 D=4:IF RND(1)<0.5 THEN D=-D
940 RETURN
950 D=2:IF RND(1)<0.5 THEN D=-D
960 RETURN
1000 POKE HH,45:POKE LL,198:POKE MM,17
1010 FOR TE=1 TO 10:NEXT
1020 POKE MM,16:RETURN
1300 GET R$:IF R$="M" THEN DR=8:RETURN
1310 IF R$="B" THEN DR=-8:RETURN
1320 IF R$="N" THEN DR=0
1330 RETURN
1500 D=255-PEEK(56321):IF D<1 OR D>2 THEN DR=0:RETURN
1510 ON D GOTO 1520,1530
1520 DR=-8:RETURN
1530 DR=8:RETURN
2000 K=0:C=53248:GOSUB 4000:ZZ=RND(TI):VV=54296:MM=54276:AA=54277:SS=54278
2010 HH=54273:LL=54272:POKE VV,15:POKE AA,0:POKE SS,240
2020 T=1:F=0:BB=F:DI=2:L1=21:L2=3:L3=10
2030 POKE 53280,0:POKE 53281,0:POKE C+21,0
2100 PRINT"J":PRINT"JEU DE CASSE-BRIQUES"
2110 PRINT"■■■■■■■■■■■■■■■■VOULEZ-VOUS LES REGLES ? (O/N)";
2120 GET R$:IF R$<>"O" AND R$<>"N" THEN 2120
2130 IF R$="N" THEN 2400
```

```
2200 PRINT"■":PRINT"VOUS DISPOSEZ D'UNE RAQUETTE ET D'UNE"
2210 PRINT"BALLE POUR DETRUIRE TROIS MURS DE "
2220 PRINT"BRIQUES.■■■■■"
2230 PRINT"IL VOUS SUFFIT DE RENVOYER LA BALLE"
2240 PRINT"AVEC LA RAQUETTE ET CHAQUE REBOND"
2250 PRINT"DE LA BALLE SUR LE MUR DETRUIRA"
2260 PRINT"UNE BRIQUE."
2270 PRINT"■■SI VOUS NE RENVOYEZ PAS LA BALLE,":PRINT"VOUS LA PERDEZ."
2280 PRINT"■■VOUS NE DISPOSEZ QUE DE TROIS BALLES..."
2290 PRINT"■■POUR CONTINUER APPUYER SUR UNE TOUCHE"
2300 GET R$:IF R$="" THEN 2300
2400 PRINT"■":PRINT"QUEL MODE CHOISISSEZ-VOUS ?"
2410 PRINT"■■■■CLAVIER        : 1"
2420 PRINT"■■MANCHE A BALAI  : 2"
2430 GET MO$:MO=VAL(MO$):IF MO<>1 AND MO<>2 THEN 2430
2500 POKE 53280,6:POKE 53281,7:POKE C+21,0:PRINT"■"
2510 FOR I=55296 TO 55335:POKE I,6:NEXT
2520 FOR I= 1024 TO 1063:POKE I,224:NEXT
2530 FOR I=24 TO 34
2540 IF I/2=INT(I/2) THEN N=0:M=8:GOTO 2560
2550 N=8:M=0
2560 FOR J=1 TO 24
2570 A=J-4*INT(J/4)
2580 IF A=1 OR A=2 THEN H=M:GOTO 2600
2590 H=N
2600 E=I+40*J
2610 POKE 55296+E,H:POKE 1024+E,160
2620 NEXT
2630 IF I=26 THEN I=31
2640 NEXT
2650 FOR J=1 TO 3
2660 FOR I=55335+J TO 56255+J STEP 40:POKE I,6:NEXT
2680 FOR I=1063+J TO 1983+J STEP 40:POKE I,224:NEXT
2690 NEXT
2700 RETURN
3000 POKE 53280,0:POKE 53281,0:PRINT"■":POKE C+21,0:PRINT"■"
3010 PRINT"    VOTRE SCORE EST DE :■■■■■■"
3020 PRINT K;"  BRIQUES ABATTUES.■■■■■■"
3030 IF K<50 THEN PRINT"C'EST UN SCORE DE DEBUTANT..."
3040 IF K>=50 AND K<109 THEN PRINT"C'EST UN SCORE HONORABLE..."
3050 IF K>=109 AND K<217 THEN PRINT"C'EST UN BON SCORE..."
3060 IF K>=217 AND K<433 THEN PRINT"C'EST UN TRES BON SCORE..."
3070 IF K>=433 THEN PRINT"C'EST UN SCORE EXCELLENT.."
3080 PRINT"■■■■VOULEZ-VOUS JOUER DE NOUVEAU ? (O/N)"
3090 GET R$:IF R$<>"O" AND R$<>"N" THEN 3090
3100 IF R$="O" THEN K=0:NB=0:GOSUB 2500:GOTO 30
3110 PRINT"■":END
4000 POKE 2041,13:FOR N=0 TO 62:READ Q:POKE 832+N,Q:NEXT
4010 POKE 2042,14:FOR N=0 TO 62:READ Q:POKE 896+N,Q:NEXT
4020 POKE 2043,15:FOR N=0 TO 62:READ Q:POKE 960+N,Q:NEXT
4030 POKE C+40,9:POKE C+41,9:POKE C+42,9
4040 RETURN
5000 DATA 255,0,0,255,0,0,255,0,0,255,0,0,255,0,0,255,0,0
5010 DATA 255,0,0,255,0,0,255,0,0,255,0,0,255,0,0,255,0,0
5020 DATA 255,0,0,255,0,0,255,0,0,255,0,0,255,0,0,255,0,0
5030 DATA 255,0,0,255,0,0,255,0,0
5040 DATA 255,0,0,255,0,0,255,0,0,255,0,0,255,0,0,255,0,0
5050 DATA 255,0,0,255,0,0,0,0,0,0,0,0,0,0,0,0,0,0
5060 DATA 0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
5070 DATA 0,0,0,0,0,0,0,0,0
5080 DATA 60,0,0,126,0,0,255,0,0,255,0,0,255,0,0,255,0,0
5090 DATA 126,0,0,60,0,0,0,0,0,0,0,0,0,0,0,0,0,0
5100 DATA 0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
```

4.3 Clavier

Ce jeu permet au débutant de s'initier de manière récréative à la connaissance parfaite du clavier de son ordinateur. En effet, le principe du jeu est le suivant : votre ordinateur choisit un caractère au hasard et l'affiche. Il vous suffit de taper ce caractère suffisamment rapidement. Après dix essais la machine vous affiche le score. Pour arrêter le jeu il suffit de demander le niveau 0 (zéro).

Structure du programme

Ligne 10: nom du jeu.

Lignes 20-50: efface l'écran et affiche le mode d'emploi.

Lignes 60-80: lecture du niveau et vérifications.

Ligne 90: calcul du nombre de boucle maximun D.

Ligne 100: remise à zéro de Z.

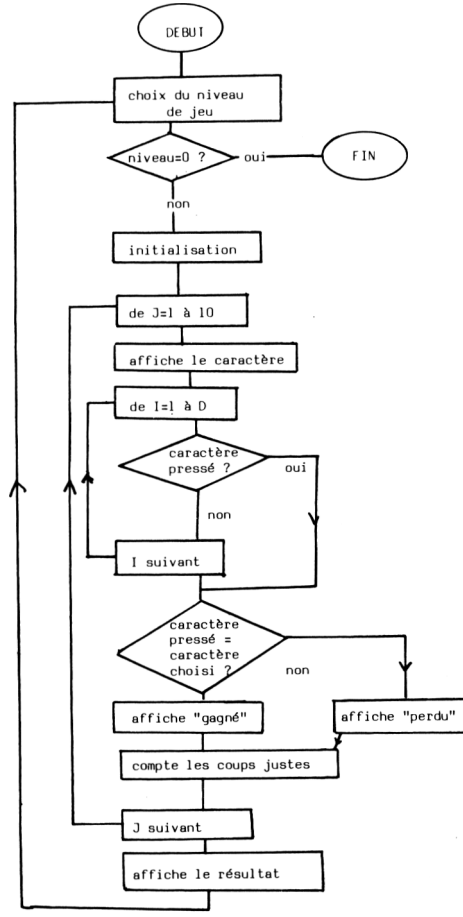
Ligne 110: début de la boucle de jeu.

Ligne 120: tirage de la lettre à chercher.

Ligne 130: immpression de cette lettre.

Lignes 140-170: boucle d'attente d'un caractère.

Ligne 180: teste si le caractère est le bon.


DÉBUT
choix du niveau de jeu
niveau=0 ?
oui
FIN
non
initialisation
de J=1 à 10
affiche le caractère
de I=1 à D
caractère pressé ?
oui
non
I suivant
caractère pressé = caractère choisi ?
non
affiche "gagné"
affiche "perdu"
compte les coups justes
J suivant
affiche le résultat

Lignes 190-210: affichent le résultat concernant ce dernier caractère et comptent le nombre de coups réussis.

Ligne 220: fin de la boucle de jeu.

Ligne 230: saut de lignes.

Ligne 240: affiche le score.

Lignes 250-260: boucle de temporisation.

Ligne 270: efface l'écran.

Ligne 280: relance le programme.

```
10 REM CLAVIER
15 PRINT"□"
20 PRINT" CONNAISSEZ VOUS VOTRE CLAVIER ?"
30 PRINT" FORCE DU JEU : 1 A 5
40 PRINT" 1 DIFICILE"
50 PRINT" 5 FACILE":PRINT"■■■      0 FIN DU JEU          ■■■■"
60 INPUT" ENTREZ VOTRE FORCE.";F
70 IF F>5 OR F<0 OR F<>INT(F) THENPRINT"ERREUR":GOTO60
80 PRINT"□":IF F=0 THEN END
90 D=100*F
100 Z=0
110 FOR J=1 TO 10
120 X=INT (RND(1)*62)+34
130 PRINTCHR$(X)
140 FOR I=1 TO D
150 GETR$
160 IF R$<>"" THEN GOTO 180
170 NEXTI
180 IF R$=CHR$(X) THEN 200
190 PRINT"  ■■ PERDU ■■"
195 GOTO 220
200 PRINT"GAGNE"
210 Z=Z+1
220 NEXTJ
230 PRINT:PRINT:PRINT:PRINT
240 PRINT" REUSSITE :";Z*10;"POUR CENT"
250 FOR I=1 TO 2000
260 NEXTI
270 PRINT"□"
280 RUN

READY.
```

4.4 Trampoline

Ce jeu est tout à fait original: tout se passe dans un cirque pendant un numéro de sauteur. Il y a deux sauteurs et une balançoire. La chute d'un sauteur sur la balançoire propulse l'autre vers le haut. Votre rôle est déterminant puisque vous devez placer la balançoire sous les sauteurs. De plus, les sauteurs, parfois, attrapent des objets, mais grande est leur maladresse.

Vous pouvez déplacer la balançoire avec le clavier dans le sens horizontal. Les touches à utiliser sont les suivantes:

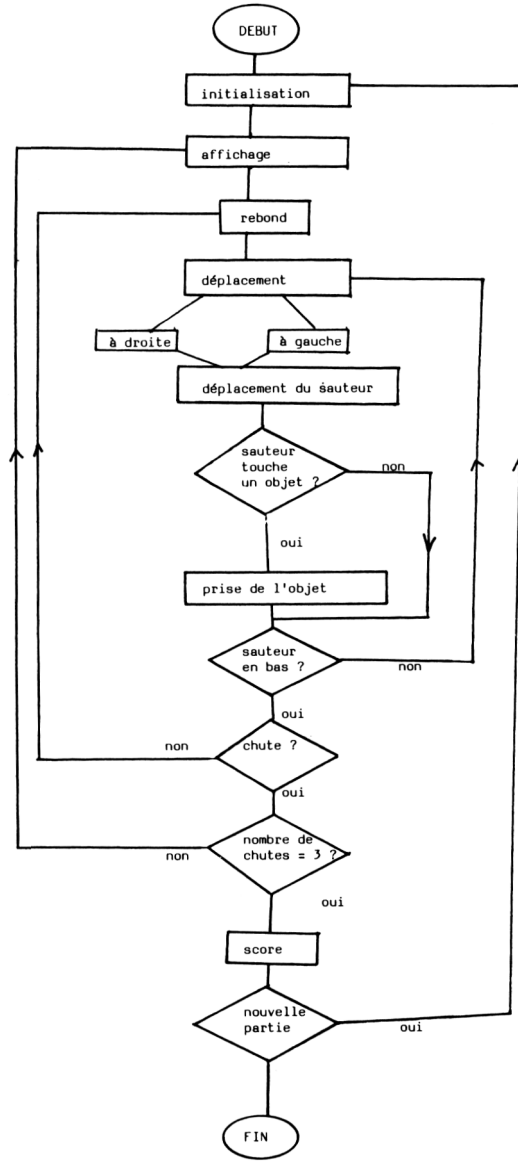
* vers la droite : touche. (point),

* vers la gauche : touche, (virgule).

Attention avant de charger ce programme il est nécessaire d'exécuter les instructions suivantes:

POKE 44, 16: POKE 16*256, 0: NEW (puis RETURN) au clavier.

En effet ce programme utilise 4 sprites différents.


DEBUT
initialisation
affichage
rebond
déplacement
à droite
à gauche
déplacement du sauteur
sauteur touche un objet ?
non
oui
prise de l'objet
sauteur en bas ?
non
oui
chute ?
non
oui
nombre de chutes = 3 ?
non
oui
score
nouvelle partie
oui
FIN

Structure du programme

Ligne 100 : appel du sous-programme d'initialisation.

Programme principal : lignes 1000 à 1130

Ligne 1000 : lecture d'un caractère au clavier.

Ligne 1010 : déplacement à gauche.

Ligne 1020 : déplacement à droite.

Lignes 1030-1040 : selon le sauteur, appel du sous-programme de déplacement de l'autre sauteur.

Lignes 1050-1060 : initialisation de l'incrément DY, et sonorisation.

Lignes 1070-1080 : changement de sauteur.

Ligne 1090 : initialisation de l'ordonnée du sauteur.

Ligne 1100 : choix aléatoire du déplacement horizontal et de la hauteur maximale que pourra atteindre le sauteur

Lignes 1110-1120 : testent si le sauteur se trouve sur les bords du terrain de jeu.

Ligne 1130 : retourne au début de la boucle.

Sous-programme de déplacement d'un sauteur: lignes 2000 à 2130

Ligne 2000: affectation des ordonnées des sprites.

Lignes 2010-2020: calcul des coordonnées du sauteur qui n'est pas sur la balançoire.

Ligne 2030-2050: contrôles du déplacement du sauteur.

Lignes 2060-2070: calcul des nouvelles coordonnées du sauteur.

Lignes 2080-2090: testent si la position du sauteur permet de prendre un objet.

Lignes 2100-2110: testent si le sauteur est tombé à coté de la balançoire.

Ligne 2120: appel du sous-programme d'initialisation.

Ligne 2130: retour au début du programme principal.

Sous-programme de déplacement de l'autre sauteur: lignes 3000 à 3130

Ce sous-programme a la même structure que le précédent.

Sous-programme de prise d'un objet: lignes 4000 à 4050

Ligne 4000: remise à zéro du registre détectant le contact sprite-fond.

Ligne 4010: teste si le sauteur peut prendre un objet.

Ligne 4020: efface l'objet.

Lignes 4030-4050: affichent le nouveau score et fin du sous-programme.

Sous-programme de fin de partie: lignes 5000 à 5130

Lignes 5000-5060: sonorisation de la fin de partie.

Lignes 5070-5090: affichent le score.

Lignes 5100-5130: fin de partie classique.

Sous-programme d'initialisation: lignes 6000 à 6340

Ligne 6000: initialise le vidéo-ship et efface l'écran.

Lignes 6010-6040: initialisation des sprites graphiques.

Ligne 6050: initialisation du compteur de chute et du score.

Lignes 6060-6100: initialisation des sprites.

Ligne 6110: initialisation relative au déplacement du sauteur.

Ligne 6120: incrémentation du nombre de chutes.

Ligne 6130: teste si le nombre de chutes est trop important pour aller en fin de partie.

Lignes 6140-6150: initialisation des coordonnées des sprites.

Ligne 6160-6330: mise en page et affichage du score.

Ligne 6340: fin du sous-programme.

Données relatives à la sonorisation et aux sprites graphiques: lignes 10000 à 10290

```
10 PRINT"■■■■■■■■ AVEZ-PENSE, AVANT DE ME CHARGER A :"
20 PRINT"■■POKE 44,16 : POKE256*16,0 : NEW■":FOR I= 1 TO 5000:NEXT
100 GOSUB6000
1000 GETR$
1010 IF R$="," THEN X=X-5 :IF X<25 THEN X=25
1020 IF R$="." THEN X=X+5 :IF X>215THEN X=215
1030 IF C=0 THEN 2000
1040 IF C=1 THEN 3000
1050 DY=-8:POKE54296,15:POKE54273,43:POKE54272,52:POKE54277,68:POKE54278,24
1060 POKE54276,129:POKE54276,128
1070 IF C=0 THEN C=1:POKEV+21,14:XA=X:GOTO1090
1080 IF C=1 THEN C=0:POKEV+21,11:XA=X+40
1090 YA=200
1100 DX=INT(RND(1)*7)-3:Q=RND(1)*50+50
1110 IF XA=25 AND DX<0 THEN DX=-DX
1120 IF XA=255 AND DX>0 THEN DX=-DX
1130 GOTO1000
2000 POKEV,X:POKEV+2,X-5
2010 XA=XA+DX
2020 YA=YA+DY
2030 IF YA<Q THEN DY=-DY
2040 IF XA<55THEN DX=0:XA=55
2050 IF XA>255THEN DX=0:XA=255
2060 POKEV+6,XA
2070 POKEV+7,YA
2080 ZZ=PEEK(V+31)
2090 IF PEEK(V+31)<>0 THEN GOSUB4000
2100 IFYA<220 THEN 1000
2110 IF X+25<XA AND X+45>XA THEN 1050
2120 GOSUB 6060
2130 GOTO 1000
3000 POKEV+4,X:POKEV+6,X+40
3010 XA=XA+DX
3020 YA=YA+DY
3030 IF YA<Q THEN DY=-DY
3040 IF XA<25THEN DX=0:XA=25
3050 IF XA>215THEN DX=0:XA=215
3060 POKEV+2,XA
3070 POKEV+3,YA
3080 ZZ=PEEK(V+31)
3090 IF PEEK(V+31)<>0 THEN GOSUB4000
3100 IFYA<220 THEN 1000
3110 IF X-10<XA AND X+10>XA THEN 1050
3120 GOSUB 6060
3130 GOTO 1000
4000 POKE V+31,0
4010 IF160=PEEK(742+40*INT((YA/8)+0.5)+INT((XA/8)+0.5))THEN NB=NB+1:DY=8
4020 POKE742+40*INT((YA/8)+0.5)+INT((XA/8)+0.5),32
4030 PRINT"■■■■■■■"
4040 PRINTTAB(35)NB;
4050 RETURN
5000 POKE54277,65:POKE54278,31
5010 RESTORE:FOR S=1 TO 15:READ N1,N2
5020 POKE54296,15
5030 POKE 54273,N1:POKE 54272,N2:POKE 54276,33
5040 FOR TE=1 TO 20:NEXT TE
5050 POKE 54276,32:FOR TE=1 TO 10:NEXT TE
5060 NEXT S:FOR I=1 TO 20:NEXT:POKE54296,32
5070 POKEV+21,0
5080 PRINT"■"
5090 PRINT"■■■■■■VOTRE SCORE TOTAL EST DE :";NB
5100 PRINT"■■■■ VOULEZ VOUS FAIRE UNE AUTRE PARTIE ?"
5110 GETR$:IF R$<>"N" ANDR$<>"O" THEN 5110
5120 IF R$="O" THEN GOSUB 6050:GOTO 1000
5130 PRINT"■":END
```

```
6000 V=53248:PRINT"□"
6010 FOR I=1 TO 30 :READN1:NEXT
6020 FOR J=0 TO 3
6030 FOR I=0 TO 62:READQ:POKE2048+I+J*64,Q:NEXTI
6040 NEXTJ
6050 CC=0:NB=0
6060 FOR J=0 TO 3
6070 POKEV+1+2*J,220
6080 POKE2040+J,32+J
6090 POKEV+39+J,2
6100 NEXTJ
6110 DX=0
6120 CC=CC+1
6130 IF CC=4 THEN GOTO5000
6140 POKEV+21,11:POKEV+29,15
6150 X=130:C=0:XA=180:YA=200:DY=-8:Q=100
6160 PRINT"□"
6170 PRINT"■                                        ■ ■"
6180 PRINT
6190 PRINTTAB(33)"■■ ■ ■■SCORE■■";
6200 PRINT"■                                        ■ ■"
6210 PRINT
6220 PRINT
6230 PRINT"■                                        ■ ■"
6240 PRINT
6250 PRINT
6260 PRINT"■                                        ■ ■■ ■NOMBRE"
6270 PRINT"                                         ■■ ■  DE  "
6280 PRINT"                                         ■■ ■CHUTES"
6290 PRINT"■":PRINTTAB(35)"■■■■■■■■■■■■■■■■■■■■";CC-1
6300 PRINT"■"
6310 FOR I=1 TO 8:PRINTTAB(33)"■■ ■■":PRINT:NEXTI
6320 PRINT"■■■■■■■■"
6330 PRINTTAB(35)NB;
6340 RETURN
10000 DATA 68,149,64,188,57,172,51,97,45,198,43,52,38,56
10010 DATA 34,75,32,94,28,214,25,177,22,227,21,154,19,63,17,27
10020 DATA 0,0,3,0,0,6,0,0,12
10030 DATA 0,0,24,0,0,48,0,0,96
10040 DATA 0,0,192,0,1,128,0,3,0
10050 DATA 0,30,0,0,60,0,0,120,0
10060 DATA 0,216,0,1,152,0,3,60,0
10070 DATA 6,60,0,12,102,0,24,102,0
10080 DATA 48,195,0,97,255,128,193,255,128
10090 DATA 60,0,0,52,0,0,62,0,0
10100 DATA 60,0,0,60,0,0,24,0,0
10110 DATA 126,0,0,126,0,0,126,0,0
10120 DATA 126,0,0,126,0,0,126,0,0
10130 DATA 126,0,0,60,0,0,28,0,0
10140 DATA 28,0,0,28,0,0,28,0,0
10150 DATA 28,0,0,28,0,0,31,0,0
10160 DATA 192,0,0,96,0,0,48,0,0
10170 DATA 24,0,0,12,0,0,6,0,0
10180 DATA 3,0,0,1,128,0,0,192,0
10190 DATA 0,120,0,0,60,0,0,30,0
10200 DATA 0,27,0,0,25,128,0,60,192
10210 DATA 0,60,96,0,102,48,0,102,24
10220 DATA 0,195,12,1,255,134,1,255,131
10230 DATA 60,0,0,44,0,0,124,0,0
10240 DATA 60,0,0,24,0,0,126,0,0
10250 DATA 126,0,0,126,0,0,126,0,0
10260 DATA 126,0,0,126,0,0,126,0,0
10270 DATA 126,0,0,60,0,0,56,0,0
10280 DATA 56,0,0,56,0,0,56,0,0
10290 DATA 56,0,0,56,0,0,248,0,0
```

4.5 Soucoupes

Vous êtes aux commandes d'un vaisseau spatial dont la mission est de raser une ville. Vous devez accomplir votre mission avec un minimum de passages et un minimum de bombes.

Le programme accepte de 1 à 4 joueurs. Au début de la partie, l'ordinateur, après avoir demandé le niveau de jeu, demande le nombre de joueurs, les noms des joueurs. Puis il affiche le nom du joueur qui commence. Quand il est prêt, celui-ci presse la barre d'espace, et le jeu commence. Le départ de la soucoupe est aléatoire de même que sa vitesse, son sens de déplacement et son altitude. La vitesse de chute de la bombe est aussi aléatoire.

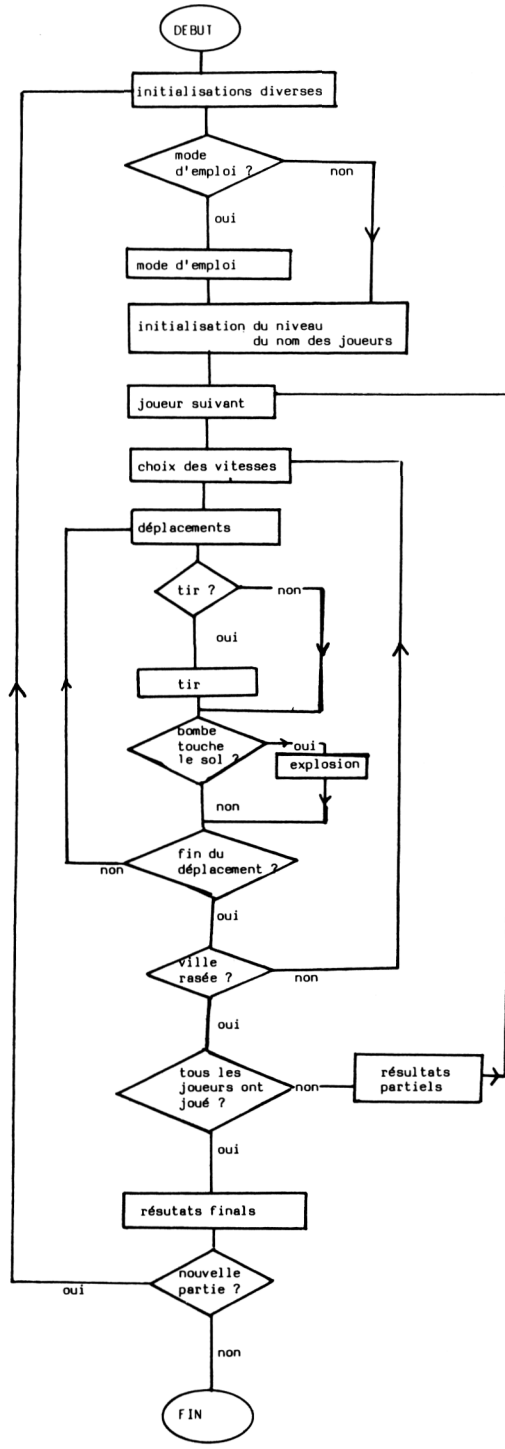
Pour lâcher une bombe, il suffit de presser la barre d'espace.

A vos manettes!!!

Structure du programme

Programme principal : lignes 1000 à 1460

Ligne 1000 : efface l'écran.


DEBUT
initialisations diverses
mode d'emploi ?
non
oui
mode d'emploi
initialisation du niveau du nom des joueurs
joueur suivant
choix des vitesses
déplacements
tir ?
non
oui
tir
bombe touche le sol ?
oui
explosion
non
fin du déplacement ?
non
oui
ville rasée ?
non
oui
tous les joueurs ont joué ?
non
résultats partiels
oui
résutats finals
nouvelle partie ?
oui
non
FIN

Ligne 1010: appel du sous-programme d'initialisation des sprites.

Ligne 1020: affiche le titre du jeu.

Ligne 1030: demande si les joueurs souhaitent lire les règles du jeu.

Ligne 1040: appel du sous-programme de réponse.

Ligne 1050: si la réponse est affirmative, appel du sous-programme de lignes.

Ligne 1060: appel du sous-programme de niveau de difficulté.

Lignes 1070-1080: demandent le nombre de joueurs.

Lignes 1090-1120: demandent les noms des joueurs.

Ligne 1130: initialisation de JI (numéro du joueur courant).

Ligne 1140: affiche le nom du joueur qui doit jouer.

Ligne 1150: appel du sous-programme qui attend que le joueur soit prêt.

Lignes 1160-1170: initialisation du nombre de passages et du nombre de coups du joueur courant.

Ligne 1180: appel du sous-programme d'affichage de la ville.

Ligne 1190: temporisation aléatoire.

Lignes 1200-1210: tirage aléatoire des vitesses des mobiles.

Ligne 1220: initialisation de l'altitude de la soucoupe.

Ligne 1230: initialisation du drapeau F.

Ligne 1240: compte le nombre de passages.

Lignes 1250-1260: tirage aléatoire du sens de déplacement de la soucoupe.

Lignes 1270-1300: testent si la ville est rasée.

Ligne 1310: si tel est le cas, retour au début de la boucle de jeu.

Ligne 1320: efface l'écran, saute quelques lignes et affiche le message correspondant.

Ligne 1330: appel du sous-programme de temporisation.

Ligne 1340: si les joueurs n'ont pas tous joué, appel du sous-programme d'affichage des résultats partiels.

Lignes 1350-1410: affichage des résultats complets.

Ligne 1420: temporisation.

Lignes 1430-1460: fin de partie classique.

Sous-programme de temporisation: lignes 2000 et 2010

Ligne 2000: boucle de temporisation.

Ligne 2010: fin du sous-programme.

Sous-programme d'attente: lignes 2500 à 2540

Ligne 2500: affiche le message corespondant.

Lignes 2510-2530: attente d'un caractère d'espace.

Ligne 2540: fin du sous-programme.

Sous-programme d'affichage des règles: lignes 3000 à 3070

Ces lignes affichent un résumé succinct des règles du jeu.

Lignes 3000-3060: affichage des règles.

Ligne 3070: appel du sous-programme de temporisation.

Ligne 3080: fin du sous-programme.

Sous-programme de choix du niveau: lignes 3500 à 3610

Lignes 3500-3540: effacent l'écran, sautent cinq lignes, affichent la valeur de chaque niveau et posent la question.

Ligne 3550: attente d'un caractère.

Ligne 3560: affecte à NV la valeur du niveau choisi.

Ligne 3570: tests sur NV pour éviter toute erreur.

Lignes 3580-3600: selon le niveau, affectent à L$ le nombre de caractères qui seront effacés à chaque explosion.

Ligne 3610: fin du sous-programme.

Sous-programme d'affichage de la ville: lignes 4000 à 4020

Ligne 4000: efface l'écran.

Ligne 4010: boucle d'affichage.

Ligne 4020: fin du sous-programme.

Sous-programme de déplacement vers la droite: lignes 4500 à 4590

Ces quelques lignes permettent à la soucoupe de se déplacer vers la droite. A chaque pas, ce sous-programme appelle le sous-programme d'affichage de la soucoupe.

Sous-programme de déplacement vers la gauche: lignes 5000 à 5090

Ce programme fait la même chose que le précédent, mais dans l'autre sens.

Sous-programme d'affichage de la soucoupe: lignes 5500 à 5610

Ligne 5500: affichage de la soucoupe.

Ligne 5510: si aucune bombe ne tombe, appel du sous-programme attente d'un tir.

Ligne 5520: sinon la bombe continue de tomber.

Ligne 5530: si la bombe n'a pas atteint la ville, va à la fin du sous-programme.

Ligne 5540: dans le cas contraire, efface la bombe.

Lignes 5550-5600: effacent la partie de la ville touchée.

Ligne 5610: fin du sous-programme.

Sous-programme de tir: lignes 6000 à 6040

Ligne 6000: teste si le tir est souhaité.

Ligne 6010: dans ce cas, mise à jour du drapeau F.

Lignes 6020-6030: calcul des coordonnées de la bombe.

Ligne 6040: fin du sous-programme.

Sous-programme d'impression des résultat partiels: lignes 6050 à 6110

Lignes 6050-6080: affichage du score.

Ligne 6090: appel du sous-programme de temporisation.

Ligne 6100: passage au joueur suivant.

Ligne 6110: fin du sous-programme.

Sous-programme de réponse: lignes 9000 à 9020

Ce sous-programme - très classique- permet d'attendre comme réponse, les seuls caractères O et N.

Sous-programme d'initialisation: lignes 9500 à 9550

Lignes 9500-9550: initialisations relatives aux sprites.

Lignes 10000-10160: données relatives aux sprites.

```
READY.


1000 PRINT"■"
1010 GOSUB9500
1020 PRINT"■■■■■■■■    L E S   S O U C O U P E S"
1030 PRINT"■■■■■■■■VOULEZ-VOUS LES REGLES DU JEU"
1040 GOSUB 9000
1050 IF R$="O" THEN GOSUB 3000
1060 GOSUB 3500
1070 INPUT"■■■■■■■■NOMBRE DE JOUEURS";J1
1080 IF J1<1 OR J1>4 OR J1<>INT(J1) THENPRINT"1 A 4 JOUEURS ":GOSUB 2000:GOTO 10
70
1090 FOR JI=1 TO J1
1100 PRINT"■NOM DU JOUEUR NUMERO ";JI
1110 INPUT J$(JI)
1120 NEXT JI
1130 JI=1
1140 PRINT "■■■■■C'EST LE TOUR DE ";J$(JI)
1150 GOSUB 2500
1160 PS(JI)=0
1170 NC(JI)=0
1180 GOSUB 4000
1190 FOR N=1 TO RND(1)*1500:NEXTN
1200 VB=RND(1)*3+1
1210 VS=RND(1)*1.5+1
1220 DP=INT(RND(1)*100) +50
1230 F=0
1240 PS(JI)=PS(JI)+1
1250 IF RND (1)<0.5 THEN GOSUB 5000:GOTO1270
1260 GOSUB4500
1270 T=0
1280 FOR J=1946 TO 1981
1290 IF PEEK(J)<>32 AND PEEK(J)<>96 THENT=1:X=1981
1300 NEXT J
1310 IF T=1 THEN 1190
1320 PRINT"■■■■■■■■■■■■■■        LA VILLE EST RASEE"
1330 GOSUB 2000
1340 PRINT"■":IF JI<J1 THEN GOSUB 6050:GOTO 1140
1350 PRINT"■■■■■■■ * * * RESULTATS COMPLETS * * *"
1360 PRINT"■■■■■■■■NOM PASSAGES COUP DE CANON  TOTAL(2P+C)"
1370 FOR JI=1 TO J1
1380 PRINTJ$(JI);
1390 PRINT TAB(7 ) PS(JI),TAB(15) NC(JI),TAB(34) 2*PS(JI)+NC(JI)
1400 PRINT
1410 NEXT JI
1420 GOSUB 2000:GOSUB2000
1430 PRINT"■■■VOULEZ VOUS FAIRE UNE AUTRE PARTIE ?"
1440 GOSUB 9000
1450 IF R$="O" THEN 1060
1460 PRINT"■":END
2000 FOR N=1 TO 1500:NEXTN
2010 RETURN
2500 PRINT"■■QUAND VOUS ETE PRETS TAPEZ SUR LA BARRE D'ESPACE"
2510 GET A$
2520 IF A$="" THEN 2510
2530 A$=LEFT$(A$,1)
2540 RETURN
3000 PRINT"■■■■  IL S'AGIT DE RASER UNE VILLE"
3010 PRINT"EN  UN MINIMUN DE PASSAGE A SA "
3020 PRINT"VERTICALE, ET AVEC  UN MINIMUN"
3030 PRINT"DE BOMBES.
```

```
3040 PRINT"■ POUR LACHER UNE BOMBE, IL"
3050 PRINT"SUFFIT DE PRESSER LA BARRE "
3060 PRINT"D'ESPACE■■■"
3070 GOSUB2500
3080 RETURN
3500 PRINT"■■■■■■NIVEAU DE DIFICULTE DU JEU"
3510 PRINT"■■ * FACILE      ──>TAPEZ 3"
3520 PRINT"■■ * MOYEN       ──>TAPEZ 2"
3530 PRINT"■■ * DIFICILE    ──>TAPEZ 1"
3540 PRINT"■■■■■■QUEL NIVEAU CHOISISSEZ-VOUS?"
3550 GET R$:IF R$=""THEN GOTO3550
3560 NV=VAL(R$)-1
3570 IF NV<0ORNV>2ORNV<>INT(NV) THEN PRINT"■ ENTRE 1,2OU 3":GOSUB 2000:GOTO3500
3580 IF NV=2 THEN L$="      "
3590 IF NV=1 THEN L$="    "
3600 IF NV=0 THEN L$=" "
3610 RETURN
4000 PRINT"□"
4010 FOR I=1946 TO 1980:POKE I,102:POKE55296+I-1024 ,1:NEXT
4020 RETURN
4500 POKEV+16,0
4510 FOR X=0 TO 255 STEPVS
4520 GOSUB 5500
4530 NEXT X
4540 POKE V+16,3
4550 FOR X=INT(X)-255 TO 64 STEP VS
4560 GOSUB 5500
4570 NEXT X
4580 POKEV+21,0
4590 RETURN
5000 POKE V+16,3
5010 FOR X=63 TO 0 STEP-VS
5020 GOSUB 5500
5030 NEXT X
5040 POKE V+16,0
5050 FOR X=255-INT(-X) TO 0 STEP-VS
5060 GOSUB 5500
5070 NEXT X
5080 POKEV+21,0
5090 RETURN
5500 POKEV+21,1+2*F:POKEV,INT(X):POKEV+1,DP
5510 IF F=0 THEN GET T$:GOSUB 6000:GOTO5610
5520 IF F=1 THEN PY=PY+VB:PX=X+3:POKEV+2,INT(X):POKEV+3,PY
5530 IF PY<230GOTO 5610
5540 POKE V+21,1
5550 POKE54296,15:POKE54277,255:POKE54278,129
5560 POKE54276,129:POKE54272,85:POKE54273,34
5570 K=X:IF PEEK(V+16)=3THENK=K+255
5580 FOR I=-NVTONV:POKE 1944-3+I+INT(K/8+0.5),32:NEXTI
5590 F=0:NC(JI)=NC(JI)+1
5600 POKE54296,0
5610 RETURN
6000 IF T$<>" "THEN 6040
6010 F=1
6020 PX=X
6030 PY=DP+3
6040 RETURN
6050 PRINT"■■■   R E S U L T A T S   P A R T I E L S"
6060 PRINT"■■■■NOMDU JOUEUR ";J$(JI)
6070 PRINT"■NOMBRES DE PASSAGES";PS(JI)
6080 PRINT"■NOMBRE DE COUP DE CANON";NC(JI)
6090 GOSUB 2500
```

```
6100 JI=JI+1
6110 RETURN
9000 GET R$
9010 IF R$<>"O" AND R$<>"N" THEN 9000
9020 RETURN
9500 V=53248
9510 POKE V+21,3 :POKEV+40,2
9520 POKE 2040,13:POKE 2041,14
9530 FOR N=0 TO 62:READ Q:POKE 832+N,Q:NEXT
9540 FOR N=0 TO 62:READ Q:POKE896+N,Q :NEXT
9550 RETURN
10000 DATA 0,255,0,3,255,192
10010 DATA 15,255,240,127,255,252,255,255,255
10020 DATA 255,255,255,255,255,255
10030 DATA 255,255,255,127,255,250
10040 DATA 15,255,240,3,255,192
10050 DATA 0,255,0,0,0,0
10060 DATA 0,0,0,0,0,0
10070 DATA 0,0,0,0,0,0
10080 DATA 0,0,0,0,0,0
10090 DATA 0,0,0,0,0,0
10100 DATA 252,0,0,252,0,0,252,0,0
10110 DATA 252,0,0,252,0,0,252,0,0
10120 DATA 120,0,0,48,0,0,48,0,0
10130 DATA 0,0,0,0,0,0,0,0,0
10140 DATA 0,0,0,0,0,0,0,0,0
10150 DATA 0,0,0,0,0,0,0,0,0
10160 DATA 0,0,0,0,0,0,0,0,0

READY.
```

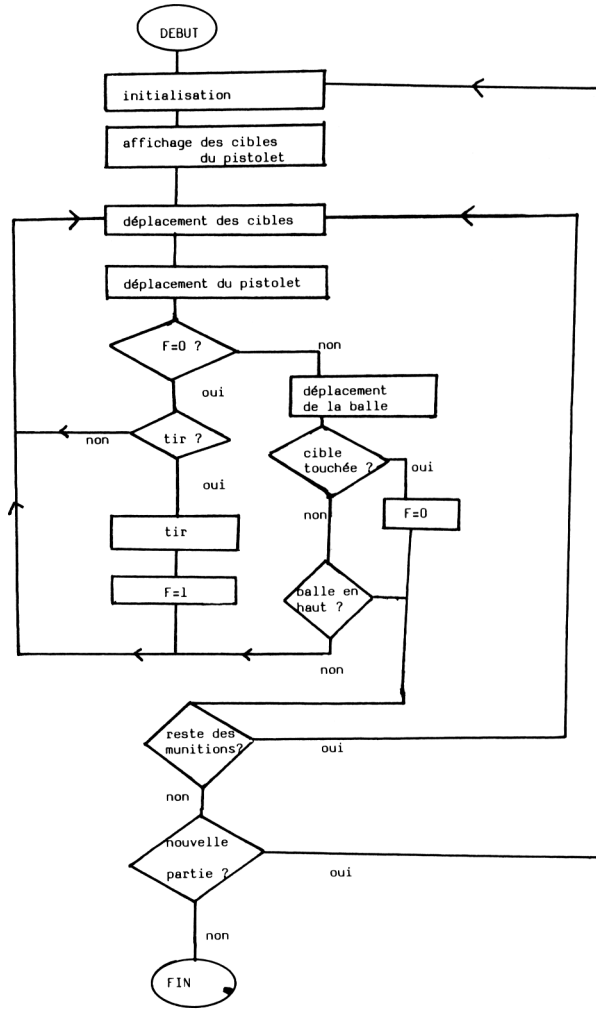
4.6 Tir sur cible

Afin de devenir tireur d'élite, il vous faut suivre un entraînement très poussé. Aux commandes de votre COMMODORE 64 vous devez tirer sur les cibles que l'ordinateur vous présentera. Pour obtenir le diplôme de tireur d'élite, tant prisé par les connaisseurs, vous devez abattre plus de vingt cibles avec les vingt-cinq cartouches que l'on vous donne. Autant dire que ce n'est pas une tâche facile. Il vous faudra beaucoup d'habileté et d'adresse.

Pour déplacer le pistolet et tirer, vous disposez du joystick. Le déplacement s'effectue horizontalement, et pour tirer, il vous suffit d'appuyer sur le bouton rouge.

Le jeu s'arrête dès que vos munitions sont épuisées, et votre score est affiché.

Pour rejouer, il suffit de presser la touche O. La touche N arrête définitivement le jeu.


DEBUT
initialisation
affichage des cibles
du pistolet
déplacement des cibles
déplacement du pistolet
F=0 ?
non
oui
déplacement
de la balle
tir ?
non
cible
touchée ?
oui
oui
F=0
non
tir
F=1
balle en
haut ?
non
reste des
munitions?
oui
non
nouvelle
partie ?
oui
non
FIN

Structure du programme

Programme principal: lignes 100 à 2100

Ligne 100: appel du sous-programme d'initialisation et de présentation.

Lignes 1010-1060: déplacement des cibles.

Ligne 2000: saisie de la position du manche à balai.

Ligne 2010: teste si le joueur a appuyé sur le bouton rouge pour tirer et si c'est le cas teste s'il a le droit de tirer.

Ligne 2020: teste si le manche à balai n'a fait aucun mouvement (il faudra éventuellement sauter la partie qui s'occupe du déplacement du tireur).

Ligne 2030: selon la position du manche à balai, permet de déplacer le tireur à gauche ou à droite.

Lignes 2040-2050: mise à jour des coordonnées du pistolet tireur suite à un déplacement à droite et test de validité.

Lignes 2060-2070: mise à jour des coordonnées du pistolet tireur suite à un déplacement à gauche et test de validité.

Ligne 2080: réactualisation des cordonnées du sprite 6 (pistolet).

Ligne 2090: teste si une balle à été tirée pour l'appel du sous-programme de déplacement de la balle.

Ligne 2100: retourne au début du programme principal.

Sous-programme de déplacement de la balle: lignes 4000 à 4100

Lignes 3000-3010: déplacement de la balle.

Lignes 3020-3030: si la balle à touché une cible, appel du sous-programme de coup au but.

Ligne 3050: teste si la balle sort de l'écran. Dans ce cas, efface, baisse le drapeau F et teste si le jeu est terminé.

Ligne 3060: fin de sous-programme.

Sous-programme de tir: lignes 4000 à 4100

Lignes 4000-4010: initialisent les coordonnées de la balle.

Ligne 4020: décrémente le nombre de balles qui restent.

Lignes 4030-4060: bruitage du tir.

Ligne 4070: effacement de la balle à droite.

Lignes 4080-4090: mise à jour des coordonnées du sprite 7 (balle)

Ligne 4100: levée du drapeau F et apparition du sprite 7.

Sous-programme de coup au but: lignes 5000 à 5120

Ligne 5000: recherche de la superposition de deux sprites.

Lignes 5010-5040: bruitage du coup au but.

Lignes 5050-5060: recherche de la cible touchée.

Ligne 5070: effacement de la cible touchée.

Ligne 5080: mise à jour du drapeau F

Ligne 5090: incrémentation du nombre de cibles touchées.

Ligne 5100: teste s'il reste des balles, sinon va en fin du jeu.

Ligne 5110: affichage de l'étoile à droite indiquant un cible touchée.

Ligne 5120: fin de sous-programme.

Sous-programme de fin de partie: lignes 6000 à 6130

Lignes 6000-6010: effacement de la page et des sprites.

Lignes 6020-6090: affichage du score et commentaire de l'instructeur (l'ordinateur).

Lignes 6100-6130: fin de partie classique.

Sous-programme d'initialisation: lignes 7000-7310

Ligne 7000: initialisation du nombre de balles.

Ligne 7010: effacement de l'écran.

Ligne 7020: initialisation du vidéo-chip.

Lignes 7030-7070: lecture des données pour les sprites graphiques.

Lignes 7080-7130: initialisation des sprites des cibles et de leurs coordonnées.

Lignes 7140-7230: initialisation des sprites du pistolet, de la balle et de leurs coordonnées.

Ligne 7240: initialisation des paramètres sonores.

Lignes 7250-7300: mise en page.

Ligne 7310: fin de sous-programme.

Lignes 10000-10200: données relative aux sprites.

```
0 REM TIR SUR CIBLE
100 GOSUB7000
1010 FOR I=0 TO 5
1020 X(I)=X(I)+3
1030 IF X(I)>255 THEN X(I)=0:POKEV+21,PEEK(V+21)+2↑I*C(I):C(I)=0
1040 POKE V+2*I,X(I)
1050 NEXTI
1060 IF F=1 THEN GOSUB 3000
2000 CB=255-PEEK(56321)
2010 IF CB>15 THEN CB=CB-16:IF F=0 THENGOSUB 4000
2020 IF CB=0 THEN GOTO2090
2030 ON CB GOTO 2080,2080,2080,2040,2080,2080,2080,2060,2080,2080
2040 X=X-3:IF X<0 THEN X=0
2050 GOTO 2080
2060 X=X+3:IF  X>255 THEN X=255
2070 GOTO 2080
2080 POKEV+12,X
2090 IF F=1 THEN GOSUB3000
2100 GOTO 1010
3000 YB=YB-4
3010 POKEV+15,YB
3020 IF YB>100 THEN GOTO3050
3030 IF PEEK(V+30)>127 THEN GOSUB5000
3040 IF YB<40 THENF=0:POKEV+21,PEEK(V+21)-128:IF B=0 THEN GOTO6000
3050 RETURN
4000 XB=X+2
4010 YB=185
4020 B=B-1
4030 FOR K=15 TO 0 STEP-1
4040 POKEVV,K:POKE W,129:POKE A,15:POKEH,40:POKEL,200
4050 NEXTK
4060 POKEW,0:POKEA,0
4070 PRINTTAB(35);"     "
4080 POKEV+15,YB
4090 POKE V+14,XB
4100 F=1:POKEV+21,PEEK(V+21)+128
4110 RETURN
5000 FOR K=15 TO 0 STEP-1
5010 POKEVV,K:POKE W,33:POKE A,9:POKEH,40:POKEL,200
5020 NEXTK
5030 POKEW,0:POKEA,0
5040 Z=PEEK(V+30):IF Z=0 THEN 5040
5050 FOR I=0 TO 5 :IF Z=128+2↑I THEN Z=I:I=5
5060 NEXTI:C(Z)=1
5070 POKE V+21,PEEK(V+21)-128-2↑Z
5080 F=0:POKEV+30,0
5090 P=P+1
5100 IF B=0 THEN 6000
5110 PRINTTAB(38);"[illegible]"
5120 RETURN
6000 PRINT"[illegible]"
6010 POKEV+21,0
6020 PRINT" [illegible]  VOUS AVEZ TOUCHE ";P;"CIBLE";
6030 IF P>1 THEN PRINT"S":PRINT
6040 PRINT" [illegible] C'EST UN SCORE D";
6050 IFP<5 THEN PRINT"E DEBUTANT":GOTO6100
6060 IFP<10 THEN PRINT"'AMATEUR":GOTO6100
6070 IFP<15 THEN PRINT"E MOYEN":GOTO6100
6080 IFP<20 THEN PRINT"E JOUEUR CONFIRME":GOTO6100
6090 PRINT"E PROFESSIONNEL"
```

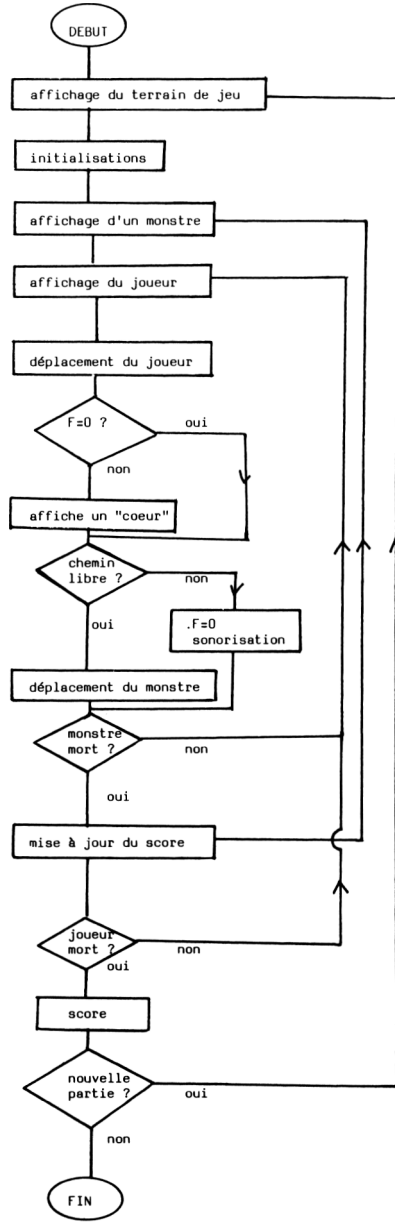
```
6100 PRINT"▒▒VOULEZ VOUS AMELIORER VOTRE SCORE ?"
6110 GETR$:IF R$<>"N" AND R$<>"O" THEN 6110
6120 IF R$="O"THEN RUN
6130 PRINT"▒":END
7000 B=25
7010 PRINT"▒";
7020 V=53248
7030 FOR I=1 TO 3
7040 FOR J=0 TO 62
7050 READQ:POKE(12+I)*64+J,Q
7060 NEXTJ
7070 NEXTI
7080 FOR I=0 TO 5
7090 POKE 2040+I,13
7100 POKEV+40+I,I+1
7110 X(I)=I*43
7120 POKE V+2*I+1,50
7130 NEXTI
7140 POKE2046,14
7150 POKE V+23,64
7160 POKE V+29,64
7170 POKE2047,15
7180 POKE V+45,5
7190 POKE V+46,5
7200 POKEV+12,160
7210 POKEV+13,200
7220 POKEV+21,127
7230 X=160
7240 VV=54296:W=54276:A=54277:H=54273:L=54272
7250 PRINT"▒";
7260 FOR I=1 TO 12:PRINT"                                    ▒ ▒ ├▒"
7270 PRINT"                                    ▒ ▒  ├▒"
7280 NEXT
7290 PRINT"                                    ▒ ▒ ├▒";
7300 PRINT"▒";
7310 RETURN
10000 DATA  0, 60,  0,  0, 66,  0,  0,129,  0
10010 DATA  1,  0,128,  1,  0,128,  1,129,128
10020 DATA  1,193,128,  1,255,128,  1,255,128
10030 DATA  1,255,128,  1,255,128,  1,255,128
10040 DATA  1,255,128,  1,255,128,  1,255,128
10050 DATA  1,255,128,  1,255,128,  1,255,128
10060 DATA  0,255,  0,  0,126,  0,  0, 60,0
10070 DATA  0, 24,  0,  0, 28,  0,  0, 24,  0
10080 DATA  0, 24,  0,  0, 24,  0,  0, 24,  0
10090 DATA  0, 24,  0,  0,124,  0,  0,124,  0
10100 DATA  0,252,  0,  1,252,  0,  2,124,  0
10110 DATA  2,252,  0,  2,124,  0,  2,252,  0
10120 DATA255,252,  0,255,248,  0,255,244,  0
10130 DATA255,236,  0,255,192,  0,255,  0,  0
10140 DATA 0, 0,  6, 0, 0, 15, 0, 0, 15
10150 DATA 0, 0, 15, 0, 0, 15, 0, 0, 15
10160 DATA 0, 0, 15, 0, 0, 15, 0, 0, 15
10170 DATA 0, 0, 15, 0, 0, 15, 0, 0, 15
10180 DATA 0, 0, 0, 0, 0, 0, 0, 0, 0
10190 DATA 0, 0, 0, 0, 0, 0, 0, 0, 0
10200 DATA 0, 0, 0, 0, 0, 0, 0, 0, 0

READY.
```

4.7 Poursuite

Vous êtes dans un plan parallèle où votre ordinateur de bord vous a égaré par erreur. Là, vous devez affronter de terribles monstres ennemis, très nombreux et remarquablement armés. Votre vaisseau ayant été endommagé par le changement d'espace-temps n'est pas en mesure de se servir de ses armes ultramodernes. IL ne vous reste qu'une solution archaïque, celle de semer des mines dans l'espace. Votre ordinateur de bord vous empêche d'aller exploser sur vos propres mines, mais il ne lui est pas possible d'amorcer une infinité de lignes minées. Il ne peut au début, semer que 10 lignes minées, ces lignes pouvant être infinies mais toute erreur de manoeuvre conduit à la rupture de ces lignes. Une erreur de manoeuvre peut être la tentatique d'aller sur une mine ou de sortir de l'espace qui vous est imparti.

Concrètement, il vous suffit de brancher le manche à balai dans le port un et de lancer le programme. Vous verrez alors un monstre ennemi vous poursuivre. Il a la formme d'un carreau. Quant à vous, pour apparaître il vous suffit de vous déplacer avec le manche à balai. Pour commencer à semer une ligne de mines, il vous suffit, de presser le bouton rouge. Vous remarquerez que le nombre de lignes de mines vous restant est affiché à droite.

Maintenant à vous de jouer, il ne vous reste plus qu'à survivre le plus longtemps possible.


DEBUT
affichage du terrain de jeu
initialisations
affichage d'un monstre
affichage du joueur
déplacement du joueur
F=0 ?
oui
non
affiche un "coeur"
chemin libre ?
non
oui
.F=0 sonorisation
déplacement du monstre
monstre mort ?
non
oui
mise à jour du score
joueur mort ?
non
oui
score
nouvelle partie ?
oui
non
FIN

Structure du programme

programme principal: lignes 1000 à 1340

Ligne 1000: efface l'écran.

Lignes 1010-1260: affichage du terrain de jeu.

Ligne 1270: initialisation des coordonnées du vaisseau du joueur.

Ligne 1280: initialisation des variables relatives au score.

Ligne 1290: initialisation du nombre de lignes de mines.

Ligne 1300: affiche le nombre des lignes restantes (B).

Ligne 1310: initialisation en vue de mesurer le temps de jeu.

Lignes 1320-1340: choix aléatoire du lieu de départ du monstre ennemi.

Sous-programme de déplacement du vaiseau du joueur: lignes 2000 à 2200

Ligne 2000: mémorisation des anciennes coordonnées du vaiseau.

Lignes 2010-2020: affichage du temps.

Ligne 2030: lecture de la position du manche à balai.

Ligne 2040: teste si le bouton rouge à été pressé, et si tel est le cas, appel du sous-programme de lancement d'une ligne de mines.

Ligne 2050: si le manche à balai est en position neutre, on saute la partie concernant le déplacement du joueur.

Lignes 2060-2140: selon la position du manche à balai, met à jour les coordonnées du joueur.

Lignes 2150-2170: selon l'indicateur F, le joueur sème un coeur.

Ligne 2180: si la case où va le joueur n'est pas libre, il revient en arrière et remet F à 0. Appel du sous-programme de sonorisation.

Lignes 2190-2200: affichent le point représentant le joueur.

Sous-programme de déplacement du monstre: lignes 3000 à 3240

Lignes 3000-3010: détermination de la direction qui rapproche le plus du joueur.

Ligne 3020: efface le monstre.

Lignes 3030-3040: mise à jour des coordonnées du monstre.

Ligne 3050: affecte à P le contenu de la case où va le monstre.

Ligne 3060: si la case est vide le monstre s'y rend.

Ligne 3070: si le joueur s'y trouve, appel du sous-programme de fin de partie.

Ligne 3080: si la case est empoisonnée, appel du sous-programme de génération d'un autre monstre.

Lignes 3090-3100: dans tout autre cas, il essaie de se déplacer vers une case voisine libre.

Lignes 3110-3180: déplacement aléatoire du monstre.

Ligne 3190: affectation à P du contenu de la case où désire se rendre le monstre.

Lignes 3200-3230: selon le contenu de la case où le monstre va, l'ordinateur appelle le sous-programme correspondant. (cf lignes 3060 à 3080).

Ligne 3240: retour au sous-programme de déplacement du joueur.

Sous-programme de génération des monstres: lignes 4000 à 4030

Lignes 4000-4020: efface le monstre et compte le nombre de monstres.

Ligne 4030: appel du programme principal pour la génération d'un monstre.

Sous-programme de fin de partie: lignes 5000 à 5140

Ligne 5000: calcul du temps écoulé.

Ligne 5010: initialisation de ZZ etX008appel007du sous-programme de sonorisation.

Ligne 5020: effacement de l'écran et saut de lignes.

Lignes 5030-5090: affichage du score.

Ligne 5100: boucle de temporisation.

Lignes 5110-5140: fin de partie classique.

Sous-programme d'initialisation d'une ligne de mine: lignes 6000 à 6070

Ligne 6000: si ce n'est pas possible retour au programme appelant.

Ligne 6010: décrémente le nombre de coups restants.

Ligne 6020: retour du curseur en haut à gauche de l'écran.

Ligne 6030: affiche le nombre de coups restants.

Lignes 6040-6050: affichent le coeur.

Ligne 6060: mise à 1 de l'indicateur F.

Ligne 6070: fin du sous-programme.

Sous-programme de sonorisation: lignes 7000 à 7070

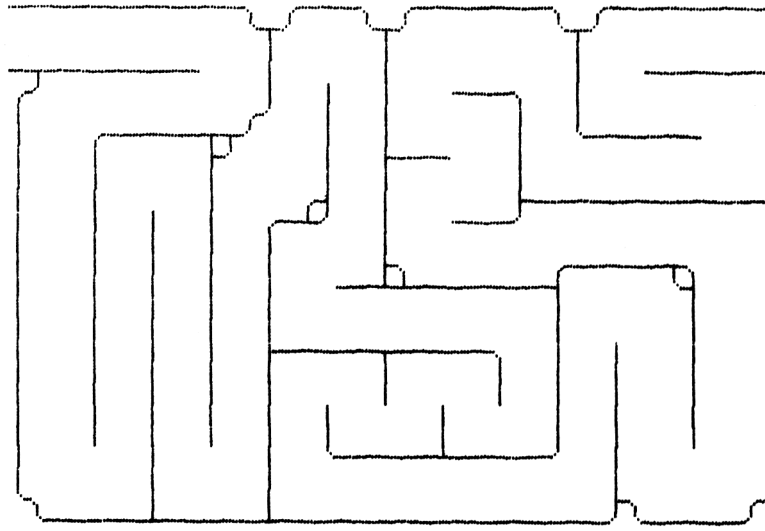
Ces lignes très classiques permettent de jouer les notes dont les valeurs se trouvent dans les lignes de DATA.

Lignes 10000 et 10010: données relatives aux notes jouées par le sous-programme de sonorisation.

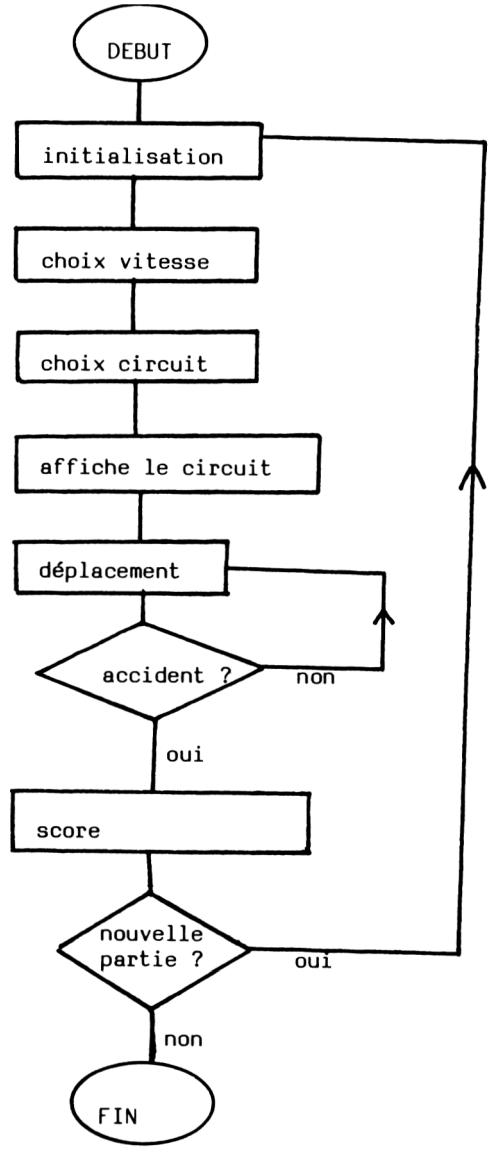
```
READY.


0 REM POURSUITE
1000 PRINT"■"
1010 PRINT"                                        "
1020 PRINT"                                        "
1030 PRINT"                                        "
1040 PRINT"                                        "
1050 PRINT"                                        "
1060 PRINT"                                        "
1070 PRINT"                                        "
1080 PRINT"                                        "
1090 PRINT"                                        "
1100 PRINT"                                        "
1110 PRINT"                                        "
1120 PRINT"                                        "
1130 PRINT"                                        "
1140 PRINT"                                        "
1150 PRINT"                                        "
1160 PRINT"                                        "
1170 PRINT"                                        "
1180 PRINT"                                        "
1190 PRINT"                                        "
1200 PRINT"                                        "
1210 PRINT"                                        "
1220 PRINT"                                        "
1230 PRINT"                                        "
1240 PRINT" ";
1250 FOR I=2 TO38 :PRINT"■";:NEXT
1260 PRINT"■ ";
1270 X=20:Y=12
1280 MM=0:Z=4
1290 M=0:B=5
1300 PRINT"■";:PRINTTAB(37)B
1310 TT=TI:F=0
1320 XM=INT(RND(1)*30+6)
1330 YM=INT(RND(1)*20+3)
1340 IF PEEK(XM+1024+40*YM)<>32 THEN 1320
2000 XA=X:YA=Y
2010 PRINT"■"
2020 PRINTINT((TI-TT)/10);
2030 CD=255-PEEK(56321)
2040 IF CD>15 THEN CD=CD-16:GOSUB 6000
2050 IF CD =0 THEN GOTO 3000
2060 ON CD GOTO 2070,2080,2150,2090,2100,2110,2150,2120,2130,2140
2070 Y=Y-1:GOTO 2150
2080 Y=Y+1:GOTO 2150
2090 X=X-1:GOTO 2150
2100 X=X-1:Y=Y-1:GOTO 2150
2110 X=X-1:Y=Y+1:GOTO 2150
2120 X=X+1:GOTO 2150
2130 X=X+1:Y=Y-1:GOTO 2150
2140 X=X+1:Y=Y+1:GOTO 2150
2150 IF F=0 THEN POKE1024+XA+40*YA,32:GOTO 2180
2160 POKE1024+XA+YA*40,83
2170 POKEXA+55296+40*YA,7
2180 ZZ=PEEK(1024+40*Y+X):IF ZZ<>32 AND ZZ<>96  THENX=XA:Y=YA:F=0:ZZ=1:GOSUB7000
2190 POKE1024+X+40*Y,81
2200 POKEX+55296+40*Y,2
3000 DX=SGN(X-XM)
3010 DY=SGN(Y-YM)
```

```
3020 POKE1024+XM+40*YM,32
3030 XM=XM+DX
3040 YM=YM+DY
3050 P=PEEK(1024+XM+40*YM)
3060 IF P=32 OR P=96 THEN POKE55296+XM+40*YM,7 :POKE1024+XM+40*YM,90:GOTO 3110
3070 IF P=81 THEN GOTO 5000
3080 IF P=83 THEN GOTO 4000
3090 IF PEEK(1024+XM+40*(YM-DY))=32 THEN YM=YM-DY:GOTO 3110
3100 IF PEEK(1024+XM-DX+40*YM)=32 THEN XM=XM-DX:GOTO 3110
3110 CM=INT(RND(1)*4+1)
3120 POKE1024+XM+40*YM,32
3130 ON CM GOTO 3140,3150,3160,3170
3140 DX=-1:DY=0:GOTO3180
3150 DX= 1:DY=0:GOTO3180
3160 DX=0:DY=1:GOTO3180
3170 DX=0:DY=-1:GOTO3180
3180 XM=XM+DX:YM=YM+DY
3190 P= PEEK(1024+XM+40*YM)
3200 IF P=32 OR P=96 THEN POKE55296+XM+40*YM,7 :POKE1024+XM+40*YM,90:GOTO 3240
3210 IF P=81 THEN 5000
3220 IF P=83 THEN 4000
3230 IF PEEK(1024+XM+40*(YM-DY))=32 THEN YM=YM-DY:GOTO 3110
3240 GOTO2000
4000 M=M+1
4010 POKE1024+XM+40*YM ,32
4020 IF M>100 THENB=B+Z:PRINT"■";:PRINTTAB(37)B:MM=MM+1:M=M-100:IFZ>0THEN Z=Z-1
4030 GOTO1320
5000 TT=TI-TT
5010 ZZ=15:GOSUB7000
5020 PRINT"■■■■■■■"
5030 PRINT"VOUS AVEZ ETE DEVORE PAR CE MONSTRE"
5040 PRINT" APRES ";INT(TT/10)/10;"SECONDES DE POURSUITE"
5050 PRINT:PRINT
5060 PRINT"VOUS AVEZ TUE ";100*MM+M;"MONSTRE";
5070 IF M>1 OR MM>=1  THEN PRINT"S"
5080 PRINT"■■■■■■■■■"
5090 PRINT" CE QUI DONNE UN SCORE DE ";(M+1)*TT ;"POINTS"
5100 FOR I=1 TO 1500:NEXT
5110 PRINT"■■■VOULEZ-VOUS PRENDRE VOTRE REVANCHE ?"
5120 GETR$:IF R$<>"O" AND R$<>"N" THEN GOTO5120
5130 IF R$="O" THEN GOTO 1000
5140 PRINT"■":END
6000 IFB=0 THEN 6070
6010 B=B-1
6020 PRINT"■";
6030 PRINTTAB(37)B;
6040 POKEX+1024+40*Y,83
6050 POKEX+55296+40*Y,7
6060 F=1
6070 RETURN
7000 POKE54277,65:POKE54278,24
7010 RESTORE:FOR S=1 TO ZZ:READ N1,N2
7020 POKE54296,15
7030 POKE 54273,N1:POKE 54272,N2:POKE 54276,33
7040 FOR TE=1 TO 20:NEXT TE
7050 POKE 54276,32:FOR TE=1 TO 10:NEXT TE
7060 NEXT S
7070 RETURN
10000 DATA 17,37,19,63,21,154,22,227,25,177,28,214,32,94,34,75
10010 DATA 38,126,43,52,45,198,51,97,57,172,64,188,68,149

READY.
```

4.8 Course automobile



Le programme suivant vous met aux commandes d'un bolide de Formule un. Il s'agit pour vous de mener ce bolide le plus loin possible sans accident. Traverser les rails de sécurité revient presque dans tous les cas à vous tuer, mais parfois si la chance est avec vous, un autre départ vous est donné. A vous de savoir en profiter !


DEBUT
initialisation
choix vitesse
choix circuit
affiche le circuit
déplacement
accident ?
non
oui
score
nouvelle partie ?
oui
non
FIN

Pour contrôler votre bolide il vous suffit de brancher un manche à balai (joystick) dans le port d'entrée-sortie 1 (à droite de l'unité centrale). Vous êtes maintenant aux commandes de cet engin. Après avoir choisi la vitesse et la difficulté du circuit, vous êtes seul contre la machine. Une fois parti il n'est plus possible de s'arrêter. Pour accélérer il suffit de presser le bouton rouge, et pour ralentir il suffit de le relâcher.

Structure du programme

Programme principal: lignes 100 à 150

Ligne 100: appel du sous-programme d'initialisation des sprites.

Ligne 110: appel du sous-programme de choix de la vitesse et du circuit.

Ligne 120: initialisation des variables de la course et du son.

Ligne 130: appel du sous-programme de course.

Ligne 140: appel du sous-programme d'accident.

Ligne 150: fin du programme.

Sous-programme de déplacement du véhicule: lignes 1000 à 1170

Ligne 1000: mémorisation de l'ancienne direction.

Ligne 1010: lecture de la position du manche6à balai.

Lignes 1020-1060: selon la position du manche à balai, mise à jour de la direction de déplacement.

Ligne 1070: mise à jour des coordonnées.

Lignes 1080-1120: vérifications des coordonnées et du registre 16 pour le déplacement dans la partie la plus à droite de l'écran.

Lignes 1130-1140: mise à jour des coordonnées des sprites.

Ligne 1150: teste si la voiture franchit les rails de sécurité.

Lignes 1160-1170: mise à jour des points. (ajoute un point si le joueur prend un virage qui n'est pas un demi-tour).

Ligne 1180: boucle de temporisation.

Ligne 1190: retourne au début de la boucle.

Ligne 1200: fin du sous-programme de déplacement du bolide.

Sous-programme d'accident: lignes 2000 à 2090

Ligne 2000: effacement des sprites, réinitialisation du pas de déplacemment de ces sprites en vue d'une éventuelle autre partie.

Ligne 2010: remise à 0 du volume du son et du contrôle de contact fond de l'écran avec les sprites.

Ligne 2020: effacement de la page.

Lignes 2030-2040: affichage du score.

Lignes 2050-2060: demande concernant une autre partie éventuelle.

Ligne 2070: effacement de l'écran.

Ligne 2080: si la réponse est affirmative alors on recommence.

Ligne 2090: fin du sous-programme.

Sous-programme s'occupant de la vitesse et du circuit: lignes 3000 à 3110

Ligne 3000: imprime la question relative au choix de la vitesse.

Ligne 3010: lecture au clavier de la vitesse.

Ligne 3020: calcul de la durée de la boucle de temporisation.

Ligne 3030: mise à 0 du compteur de point pour le score.

Lignes 3040-3080: choix du circuit.

Ligne 3090: teste si le premier circuit a été choisi et si tel est le cas appelle le sous-programme d'impression de ce circuit.

Ligne 3100: dans le cas contraire, appel du sous-programme d'affichage de l'autre circuit.

Ligne 3110: fin du sous-programme.

Sous-programme de sonorisation: lignes 4000 à 4050

Lignes 4000-4010: initialisation des coordonnées de la voiture.

Lignes 4020-4040: sonorisation proprement dite.

Ligne 4050: fin du sous-programme.

Sous-programme d'affichage du circuit numéro un: lignes 5000 à 5260

Lignes 5000-5250: affichage.

Ligne 5230: fin du sous-programme.

Sous-programme d'affichage du second circuit: lignes 6000 à 6240

Lignes 6000-6240: affichage.

Ligne 6250: fin du sous-programme.

Sous-programme d'initialisation des sprites graphiques: lignes 7000 à 7110

Lignes 7000-7050: lecture des données relatives aux sprites.

Ligne 7060: initialisation du vidéo-ship.

Lignes 7070-7080: initialisation des blocs de données des sprites.

Lignes 7090-7100: définition de la couleur des sprites.

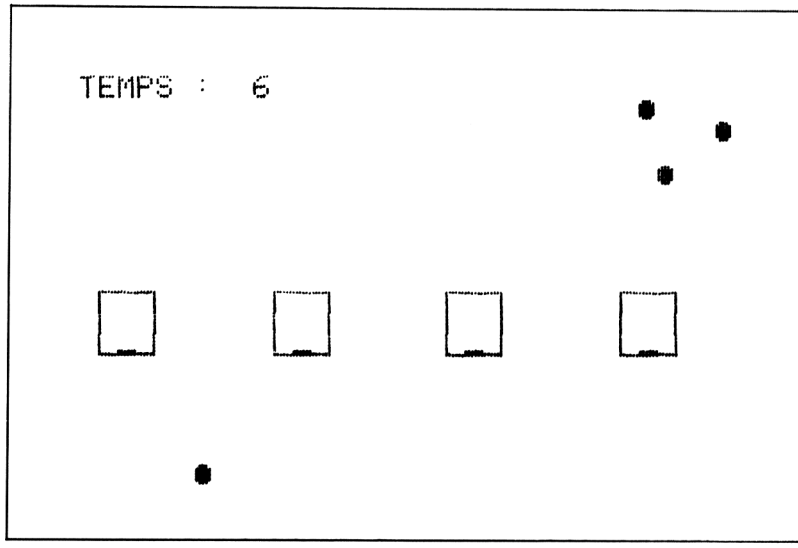
Ligne 7110: fin du sous-programme.

Données relatives aux sprites: lignes 10000 à 10130

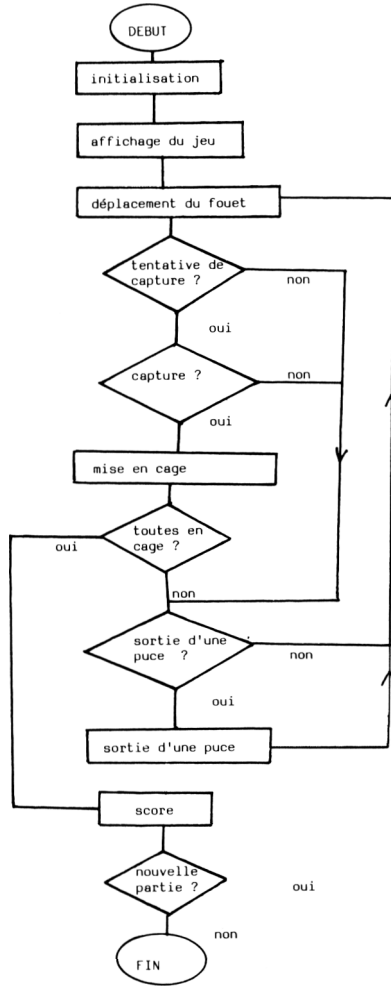
```
100 GOSUB7000
110 GOSUB  3000
120 GOSUB 4000
130 GOSUB1000
140 GOSUB 2000
150 END
1000 AN=CD:FF=0
1010 CD=255-PEEK(56321):IF CD>15 THEN CD=CD-16:FF=1
1020 IF CD=0 THEN CD=AN
1030 IF CD=2 THEN DX=0:DY=4:POKEVV+21,1 :GOTO 1070
1040 IF CD=1 THEN DX=0:DY=-4:POKEVV+21,1 :GOTO 1070
1050 IF CD=4 THEN DX=-4:DY=0:POKEVV+21,2 :GOTO 1070
1060 IF CD=8 THEN DX=4:DY=0:POKEVV+21,2 :GOTO 1070
1070 IF FF=1 THEN DX=DX*2:DY=DY*2
1080 X=X+DX:Y=Y+DY:IF X>255 THEN X=X-255:POKEVV+16,3:GOTO1110
1090 IF X<0 AND PEEK(VV+16)=3 THEN POKEVV+16,0:X=255+X:GOTO1110
1100 IF X<10 AND PEEK(VV+16)=0 THEN POKEVV+16,3:X=65+X
1110 IF Y>255 THEN Y=Y-255
1120 IF Y<0 THEN Y=255+Y
1130 POKEVV,X:POKEVV+1,Y
1140 POKEVV+2,X:POKEVV+3,Y
1150 IF PEEK(VV+31)<>0 THEN 1200
1160 IF CD<3 AND AN>3 THEN P=P+1
1170 IF CD>3 AND AN<3 THEN P=P+1
1180 FOR I=1 TO V :NEXT
1190 GOTO 1000
1200 RETURN
2000 DX=0:DY=0:CD=0:POKEVV+21,0:POKEVV+16,0
2010 POKE54296,0:POKEVV+31,0
2020 PRINT"♥"
2030 PRINT"QQQQQQQQQ          ▪ACCIDENT▪"
2040 PRINT"QQQQ   SCORE :";P
2050 PRINT"QQQVOULEZ VOUS FAIRE UN AUTRE ESSAI ?"
2060 GET R$:IF R$<>"O" AND R$<>"N" THEN 2060
2070 PRINT"♥"
2080 IF R$="O" THEN 110
2090 RETURN
3000 PRINT"♥QQQQQQQQVITESSE DE 1 A 100"
3010 INPUT V
3020 V=101-V
3030 P=0
3040 PRINT"♥"
3050 PRINT"             CHOIX DU CIRCUIT"
3060 PRINT"QQQQQQQQQQQQQQQ             1:FACILE   2:DIFFICILE"
3070 GET A$:IF A$<>"1" AND A$<>"2" THEN 3070
3080 PRINT"♥";
3090 IF A$="1" THEN GOSUB 5000:RETURN
3100 GOSUB 6000
3110 RETURN
4000 X=110
4010 Y=75
4020 POKEVV+21,1 :  POKEVV,X:POKEVV+1,Y:POKEVV+31,0
4030 POKE54296,15
4040 POKE54276,65:POKE54275,3 :POKE54274,63 :POKE54273,1  :POKE54272,18
4050 RETURN
5000 PRINT"♥";
5010 PRINT"  ╭───────────╮ ╭────────────────────╮  ";
5020 PRINT" ╱             ╲╱                      ╲ ";
5030 PRINT"|               ┤                       ╲";
5040 PRINT"|               |                        |";
5050 PRINT"|   ╭─────╮     |   ╭──────────────╮     |";
5060 PRINT"|   |     |     |   |              |     |";
5070 PRINT"|   |     |     |   |              |     |";
5080 PRINT"|   |     |     ├╮                 |     |";
```

```
5090 PRINT"
5100 PRINT"
5110 PRINT"
5120 PRINT"
5130 PRINT"
5140 PRINT"
5150 PRINT"
5160 PRINT"
5170 PRINT"
5180 PRINT"
5190 PRINT"
5200 PRINT"
5210 PRINT"
5220 PRINT"
5230 PRINT"
5240 PRINT"
5250 PRINT"
5260 RETURN
6000 PRINT"
6010 PRINT"
6020 PRINT"
6030 PRINT"
6040 PRINT"
6050 PRINT"
6060 PRINT"
6070 PRINT"
6080 PRINT"
6090 PRINT"
6100 PRINT"
6110 PRINT"
6120 PRINT"
6130 PRINT"
6140 PRINT"
6150 PRINT"
6160 PRINT"
6170 PRINT"
6180 PRINT"
6190 PRINT"
6200 PRINT"
6210 PRINT"
6220 PRINT"
6230 PRINT"
6240 PRINT"
6250 RETURN
7000 FOR I=0 TO 62
7010 READQ:POKE832+I,Q
7020 NEXTI
7030 FOR I=0 TO 62
7040 READQ:POKE896+I,Q
7050 NEXTI
7060 VV=53248
7070 POKE2040,13
7080 POKE2041,14
7090 POKE VV+39,2
7100 POKE VV+40,2
7110 RETURN
10000 DATA 24,0,0,60,0,0,189,0,0,255,0,0,189,0,0,36,0,0
10010 DATA 189,0,0,255,0,0,189,0,0,60,0,0,24,0,0,0,0,0
10020 DATA 0,0,0,0,0,0,0,0,0
10030 DATA 0,0,0,0,0,0,0,0,0
10040 DATA 0,0,0,0,0,0,0,0,0
10050 DATA 59,128,0,17,0,0,127,192,0
10060 DATA 251,224,0
10070 DATA 251,224,0
10080 DATA 127,192,0,17,0,0,59,128,0
10090 DATA 0,0,0,0,0,0,0,0,0
10100 DATA 0,0,0,0,0,0,0,0,0
10110 DATA 0,0,0,0,0,0,0,0,0
10120 DATA 0,0,0,0,0,0,0,0,0
10130 DATA 0,0,0

READY.
```

4.9 Puces savantes



Ce programme vous permet de vous lancer dans une activité pour le moins originale: le dressage de puces. Comme chacun sait, le dressage de puces demande une grande habileté. En particulier, le dresseur doit pouvoir faire rentrer son troupeau au bercail très rapidement. On sait bien que ces animaux ne sont pas très dociles. Afin de vous initier à ce jeu je vous propose de vous attacher à cette tâche ardue: vous devrez en un minimum de temps faire rentrer tous ces animaux dans leurs cages. Pour cela vous disposez d'un fouet électrique (on n'arrête pas le progrès !).


DEBUT
initialisation
affichage du jeu
déplacement du fouet
tentative de capture ?
non
oui
capture ?
non
oui
mise en cage
toutes en cage ?
oui
non
sortie d'une puce ?
non
oui
sortie d'une puce
score
nouvelle partie ?
oui
non
FIN

Pratiquement, il vous suffit de brancher le manche à balai dans le port 1 et de lancer le programme. Vous verrez apparaître 4 cages, 4 puces et le fouet électrique que vous manierez avec le joystick. °Pour faire rentrer une puce dans sa cage, il suffit de placer le fouet au-dessus d'elle et de presser le bouton rouge. La puce ainsi touchée ira en cage. °Attention, les puces osent sortir de leur cage au bout d'un certain temps !

Structure du programme

Ligne 100: va au programme principal.

Programme principal: lignes 2000 à 2720

Ligne 2000: efface l'écran.

Lignes 2010-2130: affichage des cages.

Lignes 2140-2150: affichage du temps.

Lignes 2170-2200: tirage des coordonnées des puces.

Lignes 2210-2220: coordonnées des cages des puces.

Lignes 2230-2260: affichage des puces.

Ligne 2270: initialisation de D.

Ligne 2280: initialisations relatives au temps.

Lignes 2290-2350: initialisations relatives au sprite.

Lignes 2360-2370: initialisation des coordonnées du sprite.

Ligne 2380: affichage du temps.

Ligne 2390: lecture de la position du manche à balai.

Ligne 2400: si celui-ci est en position neutre, retour au début de la boucle de déplacement.

Ligne 2410: si le bouton rouge a été pressé, appel du sous-programme de dressage.

Ligne 2420: selon la position du joystick, appel d'un sous-programme de déplacement.

Lignes 2430-2580: déplacement du fouet.

Ligne 2590: incrémentaion de T.

Lignes 2600-2710: sortie aléatoire d'une puce.

Ligne 2720: retour au début de la boucle de déplacement.

Sous-programme de dressage: lignes 1000 à 1130

Ces quelques lignes testent si une puce est à portée du fouet et si tel est le cas, la renvoie dans sa cage. De plus, elles testent si toutes les puces sont à leur place.

Sous-programme de fin de partie: lignes 3000 à 3160

Lignes 3000-3010: effacement de l'écran et du sprite.

Lignes 3020-3050: affichage du score.

Lignes 3060-3120: sonorisation.

Lignes 3130-3160: fin de partie classique.

Sous-programme de réponse: lignes 9000 à 9020

Ce sous-programme permet d'attendre que le joueur tape un 0 ou un I, en inhibant tout autre caractère.

Ligne 9000: saisie d'un caractère au clavier.

Ligne 9010: teste si ce caractère est un de ceux souhaités.

Ligne 9020: fin du sous-programme (retour au programme appelant).

Données relatives à la sonorisation: lignes 10000 & 10010

Données relatives au sprite: lignes 10020 à 10080

```
100 GOTO2000
1000 FOR I=1 TO 4
1010 IF Y-50>Y(I)*8 ORY(I)*8>Y+21-50THEN 1120
1020 XX=X-30:IFPEEK(V+16)>=4 THEN  XX=X+225
1030 IF XX>X(I)*8 ORX(I)*8>XX+21 THEN 1120
1040 POKE 1024+X(I)+40*Y(I),32
1050 X(I)=XX(I):Y(I)=YY(I)
1060 D=D-1
1070 POKE54296,15:POKE54273,36:POKE54272,75:POKE54276,17:POKE54276,16
1080 POKE 1024+X(I)+40*Y(I),81
1090 POKE 55296+40*Y(I)+X(I),2
1100 IF D=0 THEN 3000
1110 I=4:POKE54296,0
1120 NEXTI
1130 RETURN
2000 PRINT"□"
2010 FORI=1TO 10 :PRINT:NEXT
2020 FOR I=1 TO4
2030 PRINT"     ┌──┐ ";
2040 NEXTI
2050 PRINT
2060 FORI=1 TO 4
2070 PRINT"     │  │ ";
2080 NEXTI
2090 PRINT
2100 FOR I=1 TO 4
2110 PRINT"     └──┘ ";
2120 NEXT I
2130 PRINT
2140 PRINT"■"
2150 PRINT"    TEMPS : ";
2160 FOR I=1 TO 4
2170 Y(I)=INT(RND(1)*25)
2180 IF Y(I)>10 AND Y(I)<14 THEN 2170
2190 X(I)=INT(RND(1)*40)
2200 NEXTI
2210 XX(1)=6   :XX(2)=15  :XX(3)=24  :XX(4)=33
2220 FOR I=1 TO 4 :YY(I)=12:NEXT
2230 FOR I=1 TO 4
2240 POKE 1024+X(I)+40*Y(I),81
2250 POKE 55296+40*Y(I)+X(I),2
2260 NEXTI
2270 D=4
2280 T=0:TT=TI
2290 V=53248
2300 POKEV+21,4:POKEV+41,2
2310 POKE 2042,13
2320 FOR I= 1 TO 30:READQ:NEXT
2330 FOR N=0 TO 62 : READQ:POKE 832+N,Q
2340 NEXT N
2350 POKEV+16,0
2360 X=160:Y=100
2370 POKEV+4,X:POKEV+5,Y
2380 PRINT"■":PRINTTAB(12) INT((TI-TT)/10)
2390 CD=255-PEEK(56321)
2400 IF CD=0THEN 2370
2410 IF CD>15 THENCD=CD-16:GOSUB 1000
2420 ON CD GOTO 2460,2450,2440,2470,2510,2490,2440,2530,2570,2550
2430 GOTO 2370
2440 GOTO 2590
2450 Y=Y+6:GOTO 2590
2460 Y=Y-6:GOTO 2590
```

```
2470 X=X-6:IF PEEK(V+16)>=4 ANDX<0 THEN POKEV+16,0:X=255+X
2480 GOTO 2590
2490 X=X-6:Y=Y+6:IF PEEK(V+16)>=4 ANDX<0 THEN POKEV+16,0:X=255+X
2500 GOTO 2590
2510 X=X-6:Y=Y-6:IF PEEK(V+16)>=4 ANDX<0 THEN POKEV+16,0:X=255+X
2520 GOTO 2590
2530 X=X+6:IF X>255THENX=X-255:POKEV+16,4
2540 GOTO 2590
2550 X=X+6:Y=Y+6:IF X>255THENX=X-255:POKEV+16,4
2560 GOTO 2590
2570 X=X+6:Y=Y-6:IF X>255THENX=X-255:POKEV+16,4
2580 GOTO 2590
2590 T=T+1
2600 IF T<25*D THENGOTO 2720
2610 T=0
2620 I=INT(RND(1)*4+1)
2630 IFPEEK(1024+40*YY(I)+XX(I))<>81 THEN  GOTO 2720
2640 POKE 1024+40*Y(I)+X(I),32
2650 Y(I)=INT(RND(I)*23)+2
2660 IF Y(I)>10 AND Y(I)<14 THEN 2650
2670 X(I)=INT(RND(1)*40)
2680 POKE 1024+40*Y(I)+X(I),81
2690 POKE 55296+40*Y(I)+X(I),2
2700 D=D+1
2710 IF D=0 THENGOTO 3000
2720 GOTO 2370
3000 POKE V+21,0
3010 PRINT"□"
3020 FOR I=1 TO 10 :PRINT:NEXT
3030 PRINT"   B R A V O !":PRINT:PRINT
3040 PRINT" VOUS AVEZ REUSSI A DRESSER CES PUCES"
3050 PRINT:PRINT"        EN ";(INT((TI-TT)/10))/10;" SECONDES"
3060 POKE54277,65:POKE54278,31
3070 RESTORE:FOR S=1 TO 15:READ N1,N2
3080 POKE54296,15
3090 POKE 54273,N1:POKE 54272,N2:POKE 54276,33
3100 FOR TE=1 TO 20:NEXT TE
3110 POKE 54276,32:FOR TE=1 TO 10:NEXT TE
3120 NEXT S
3130 PRINT:PRINT:PRINT:PRINT"VOULEZ-VOUS FAIRE UNE AUTRE PARTIE ?"
3140 GOSUB9000:POKE54296,0
3150 IF R$="O" THEN RUN
3160 PRINT"□":END
9000 GETR$
9010 IFR$<>"O" ANDR$<>"N" THEN  GOTO 9000
9020 RETURN
10000 DATA 17,37,19,63,21,154,22,227,25,177,28,214,32,94,34,75
10010 DATA 38,126,43,52,45,198,51,97,57,172,64,188,68,149
10020 DATA255,255,255,128,0,1,128,0,1
10030 DATA128,0  ,1  ,128,0,1,128,0,1
10040 DATA128,0  ,1  ,128,0,1,128,0,1
10050 DATA128,0  ,1  ,128,0,1,128,0,1
10060 DATA128,0  ,1  ,128,0,1,128,0,1
10070 DATA128,0  ,1  ,128,0,1,128,0,1
10080 DATA128,0  ,1  ,128,0,1,255,255,255

READY.
```

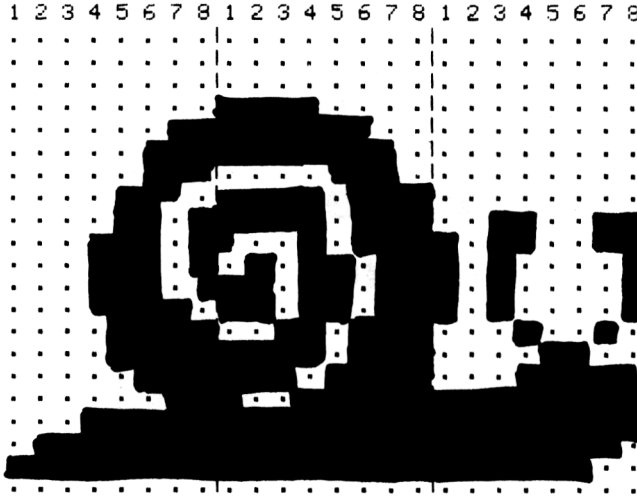
Annexe A

Les Sprites graphiques

1. Sprites graphiques

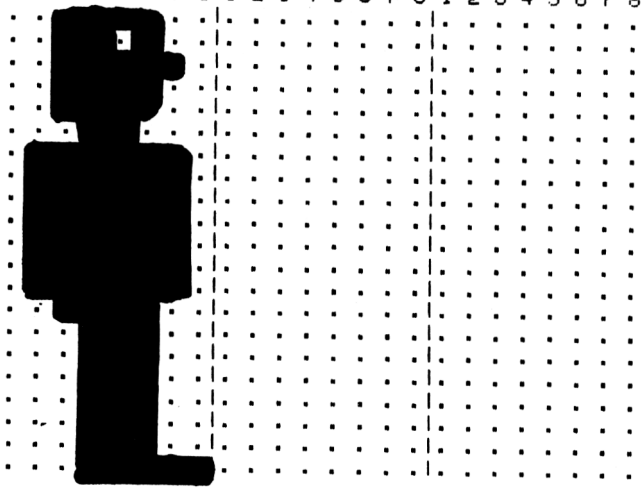
Vous trouverez dans cette annexe tous les sprites définis dans l'ensemble des programmes de ce recueil, sous forme de schémas, ainsi que les valeurs à mettre dans les lignes de données (DATA) correspondantes.

Course d'escargots

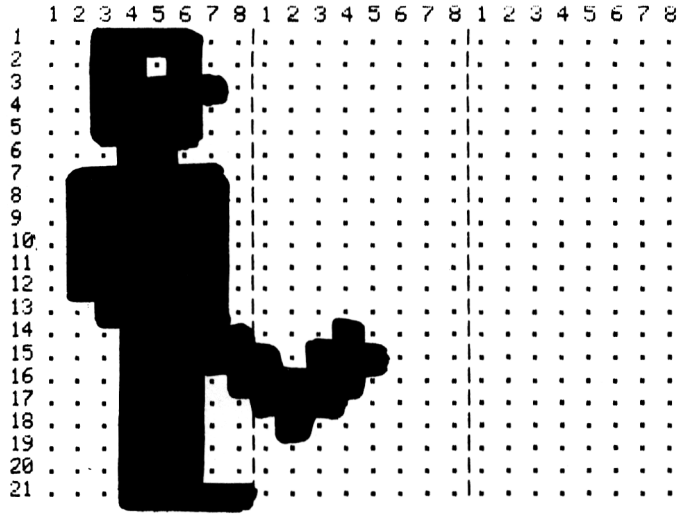


1	0	0	0
2	0	0	0
3	0	0	0
4	0	240	0
5	3	252	0
6	7	254	0
7	7	14	0
8	14	247	0
9	13	247	51
10	29	151	179
11	29	91	161
12	29	219	161
13	30	219	161
14	15	55	18
15	15	247	12
16	7	239	31
17	3	159	255
18	31	255	255
19	127	255	254
20	255	255	252
21	0	0	0

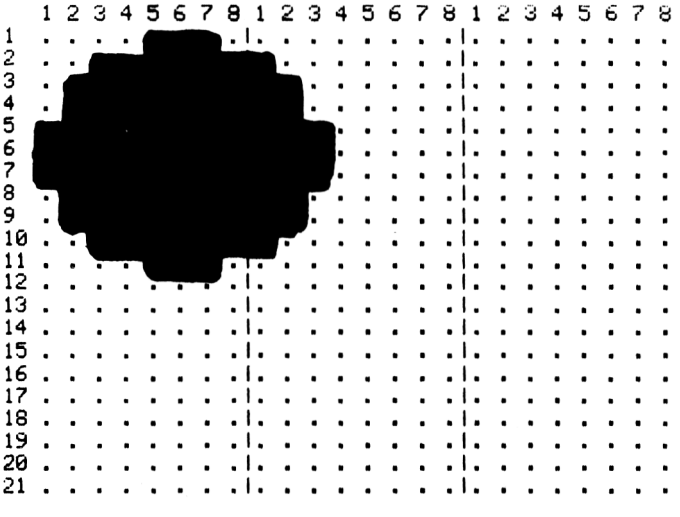
Balle au pied



1	60	0	0
2	52	0	0
3	62	0	0
4	60	0	0
5	60	0	0
6	24	0	0
7	126	0	0
8	126	0	0
9	126	0	0
10	126	0	0
11	126	0	0
12	126	0	0
13	126	0	0
14	60	0	0
15	28	0	0
16	28	0	0
17	28	0	0
18	28	0	0
19	28	0	0
20	28	0	0
21	31	0	0

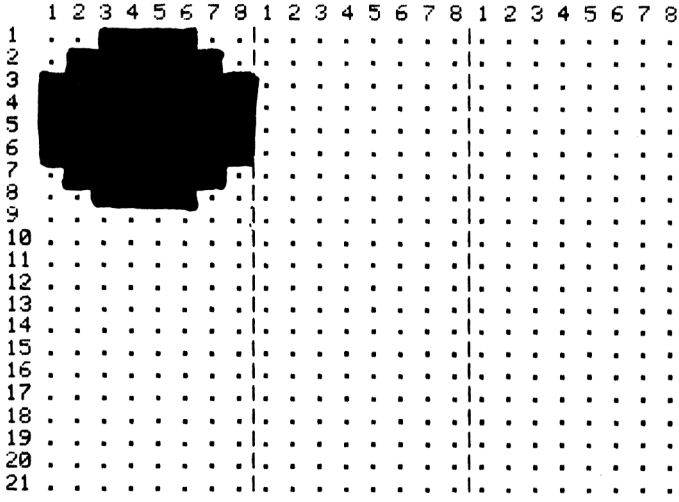


1	60	0	0
2	52	0	0
3	62	0	0
4	60	0	0
5	60	0	0
6	24	0	0
7	126	0	0
8	126	0	0
9	126	0	0
10	126	0	0
11	126	0	0
12	126	0	0
13	60	0	0
14	31	16	0
15	31	56	0
16	29	240	0
17	28	224	0
18	28	64	0
19	28	0	0
20	28	0	0
21	31	0	0

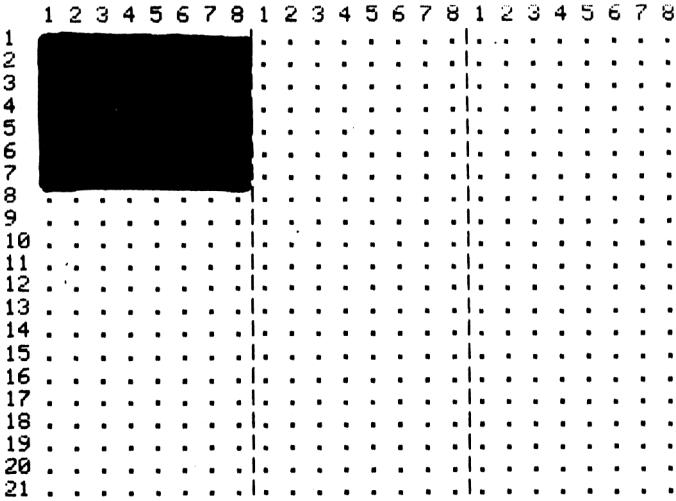


1	14	0	0
2	63	128	0
3	127	192	0
4	127	192	0
5	255	224	0
6	255	224	0
7	255	224	0
8	127	192	0
9	127	192	0
10	63	128	0
11	14	0	0
12	0	0	0
13	0	0	0
14	0	0	0
15	0	0	0
16	0	0	0
17	0	0	0
18	0	0	0
19	0	0	0
20	0	0	0
21	0	0	0

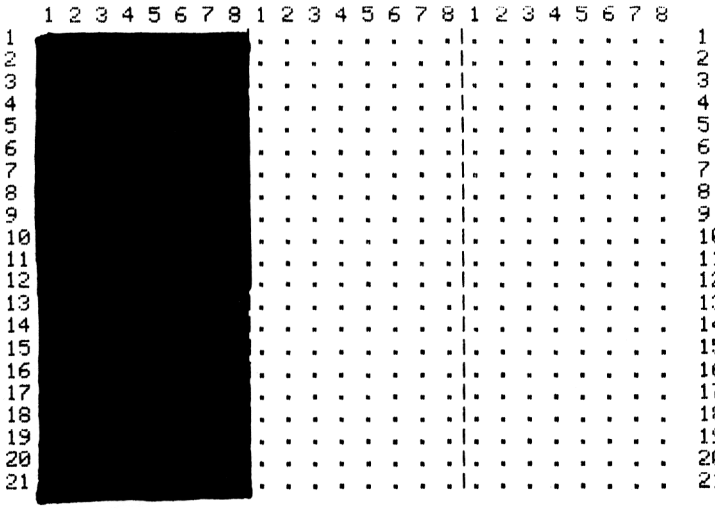
Squash et mur de brique



1	60	0	0
2	126	0	0
3	255	0	0
4	255	0	0
5	255	0	0
6	255	0	0
7	126	0	0
8	60	0	0
9	0	0	0
10	0	0	0
11	0	0	0
12	0	0	0
13	0	0	0
14	0	0	0
15	0	0	0
16	0	0	0
17	0	0	0
18	0	0	0
19	0	0	0
20	0	0	0
21	0	0	0

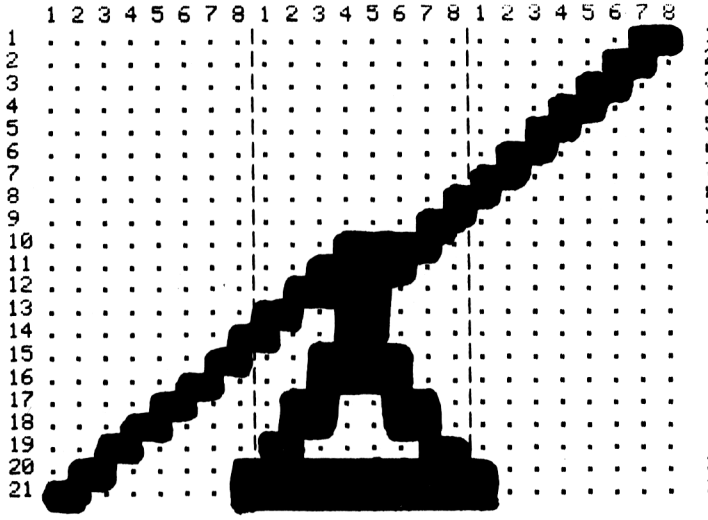


1	255	0	0
2	255	0	0
3	255	0	0
4	255	0	0
5	255	0	0
6	255	0	0
7	255	0	0
8	0	0	0
9	0	0	0
10	0	0	0
11	0	0	0
12	0	0	0
13	0	0	0
14	0	0	0
15	0	0	0
16	0	0	0
17	0	0	0
18	0	0	0
19	0	0	0
20	0	0	0
21	0	0	0

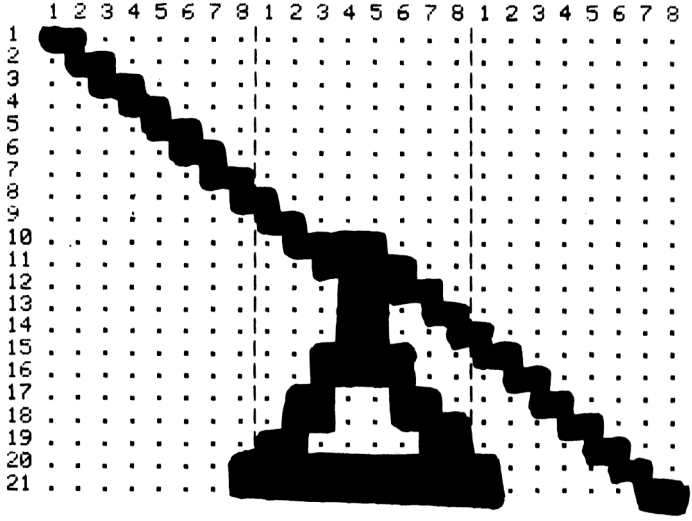


255	0	0
255	0	0
255	0	0
255	0	0
255	0	0
255	0	0
255	0	0
255	0	0
255	0	0
255	0	0
255	0	0
255	0	0
255	0	0
255	0	0
255	0	0
255	0	0
255	0	0
255	0	0
255	0	0
255	0	0
255	0	0

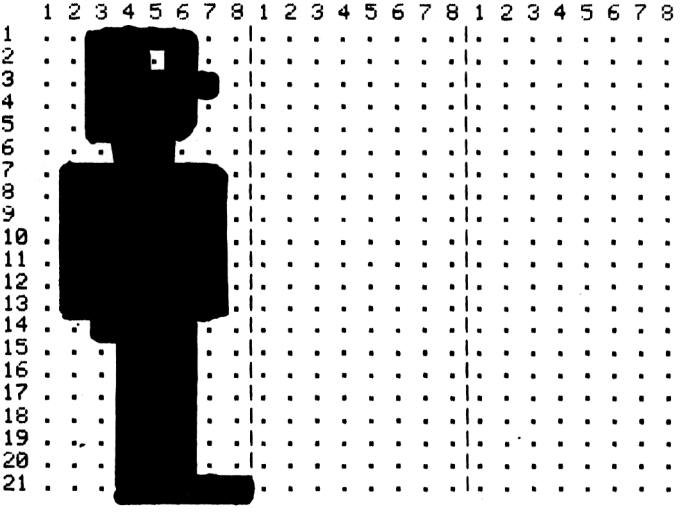
Trampoline



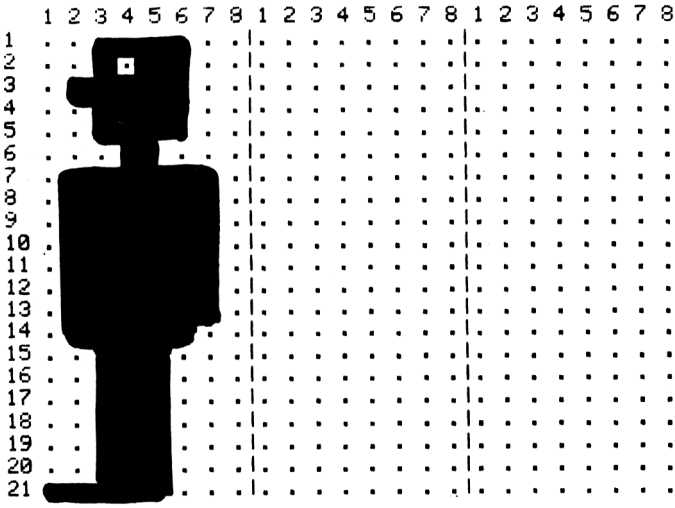
0	0	3
0	0	6
0	0	12
0	0	24
0	0	48
0	0	96
0	0	192
0	1	128
0	3	0
0	30	0
0	60	0
0	120	0
0	216	0
1	152	0
3	60	0
6	60	0
12	102	0
24	102	0
48	195	0
97	255	128
193	255	128



1	192	0	0
2	96	0	0
3	48	0	0
4	24	0	0
5	12	0	0
6	6	0	0
7	3	0	0
8	1	128	0
9	0	192	0
10	0	120	0
11	0	60	0
12	0	30	0
13	0	27	0
14	0	25	128
15	0	60	192
16	0	60	96
17	0	102	48
18	0	102	24
19	0	195	12
20	1	255	134
21	1	255	131

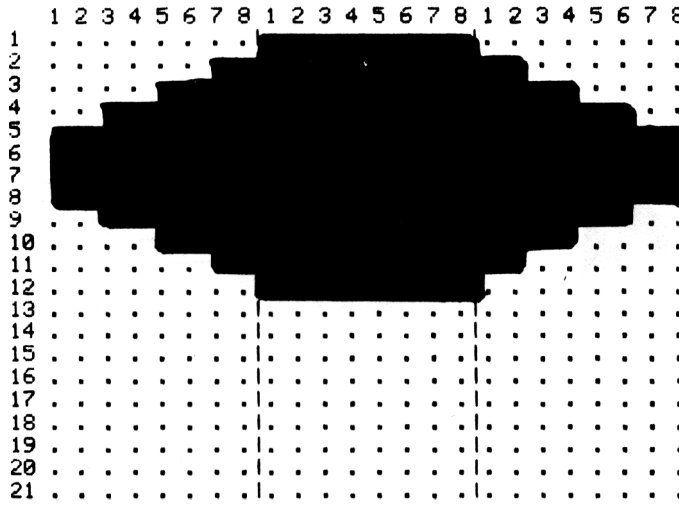


1	60	0	0
2	52	0	0
3	62	0	0
4	60	0	0
5	60	0	0
6	24	0	0
7	126	0	0
8	126	0	0
9	126	0	0
10	126	0	0
11	126	0	0
12	126	0	0
13	126	0	0
14	60	0	0
15	28	0	0
16	28	0	0
17	28	0	0
18	28	0	0
19	28	0	0
20	28	0	0
21	31	0	0

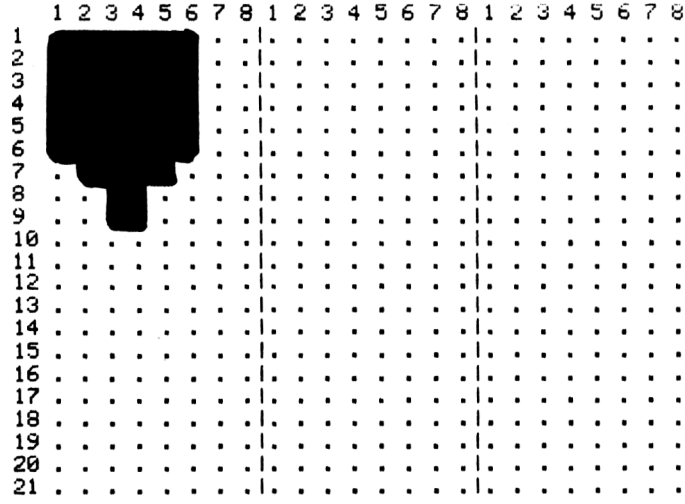


1	0	255	0
2	3	255	192
3	15	255	240
4	127	255	252
5	255	255	255
6	255	255	255
7	255	255	255
8	255	255	255
9	127	255	252
10	15	255	240
11	3	255	192
12	0	255	0
13	0	0	0
14	0	0	0
15	0	0	0
16	0	0	0
17	0	0	0
18	0	0	0
19	0	0	0
20	0	0	0
21	0	0	0

Soucoupes

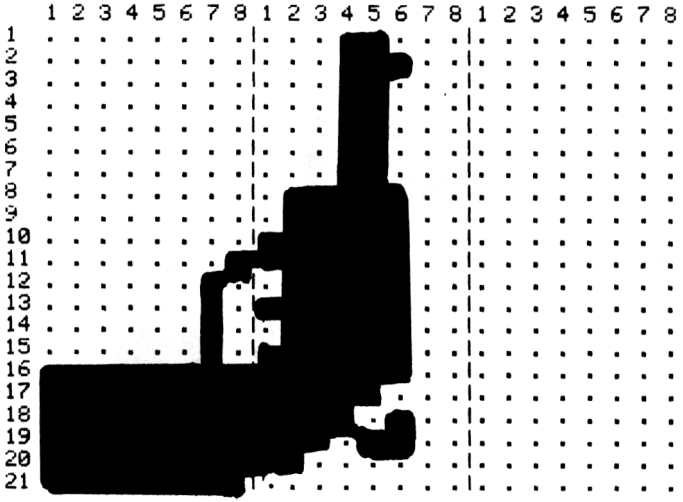


1	60	0	0
2	44	0	0
3	124	0	0
4	60	0	0
5	60	0	0
6	24	0	0
7	126	0	0
8	126	0	0
9	126	0	0
10	126	0	0
11	126	0	0
12	126	0	0
13	126	0	0
14	60	0	0
15	56	0	0
16	56	0	0
17	56	0	0
18	56	0	0
19	56	0	0
20	56	0	0
21	248	0	0

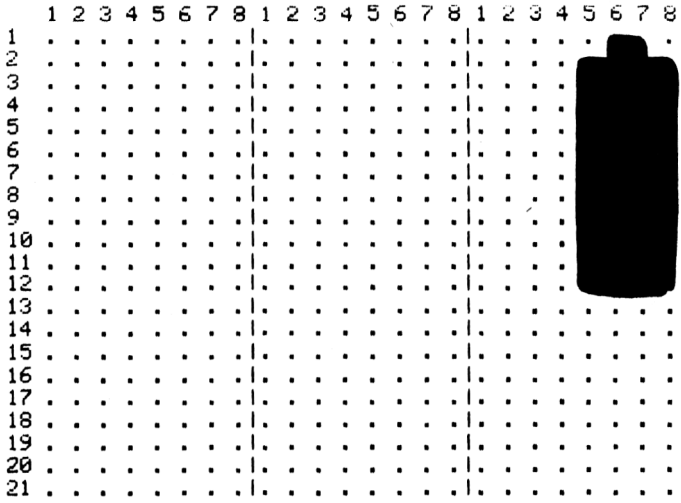


1	254	0	0
2	254	0	0
3	254	0	0
4	254	0	0
5	254	0	0
6	254	0	0
7	120	0	0
8	48	0	0
9	48	0	0
10	0	0	0
11	0	0	0
12	0	0	0
13	0	0	0
14	0	0	0
15	0	0	0
16	0	0	0
17	0	0	0
18	0	0	0
19	0	0	0
20	0	0	0
21	0	0	0

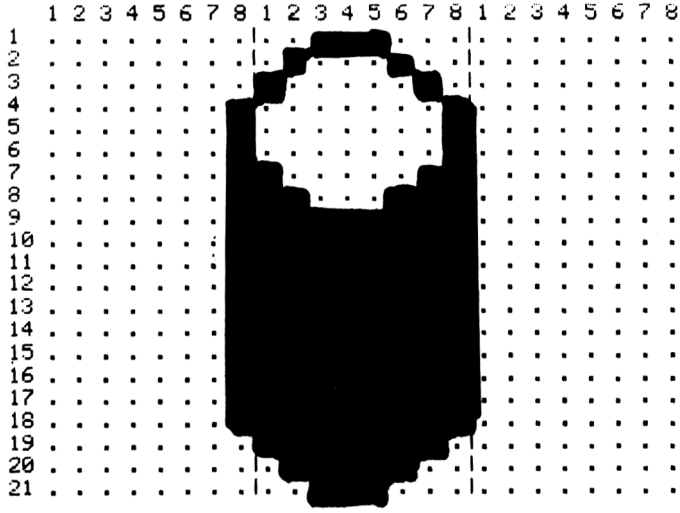
Tir sur cible



1	0	24	0
2	0	28	0
3	0	24	0
4	0	24	0
5	0	24	0
6	0	24	0
7	0	24	0
8	0	124	0
9	0	124	0
10	0	252	0
11	1	252	0
12	2	124	0
13	2	252	0
14	2	124	0
15	2	252	0
16	255	252	0
17	255	248	0
18	255	244	0
19	255	236	0
20	255	192	0
21	255	0	0

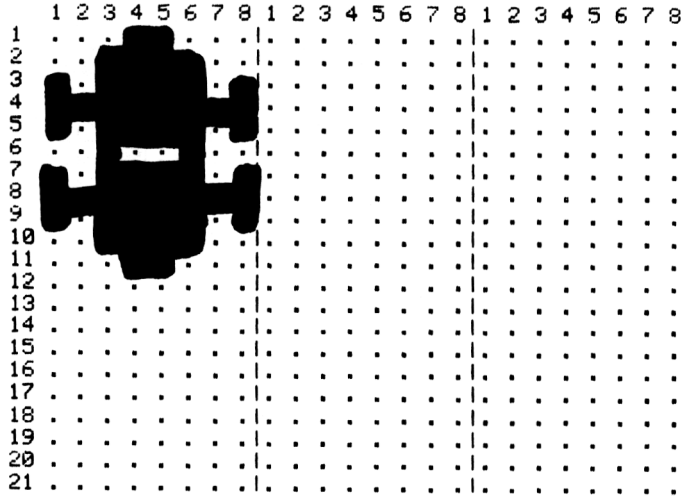


1	0	0	6
2	0	0	15
3	0	0	15
4	0	0	15
5	0	0	15
6	0	0	15
7	0	0	15
8	0	0	15
9	0	0	15
10	0	0	15
11	0	0	15
12	0	0	15
13	0	0	0
14	0	0	0
15	0	0	0
16	0	0	0
17	0	0	0
18	0	0	0
19	0	0	0
20	0	0	0
21	0	0	0

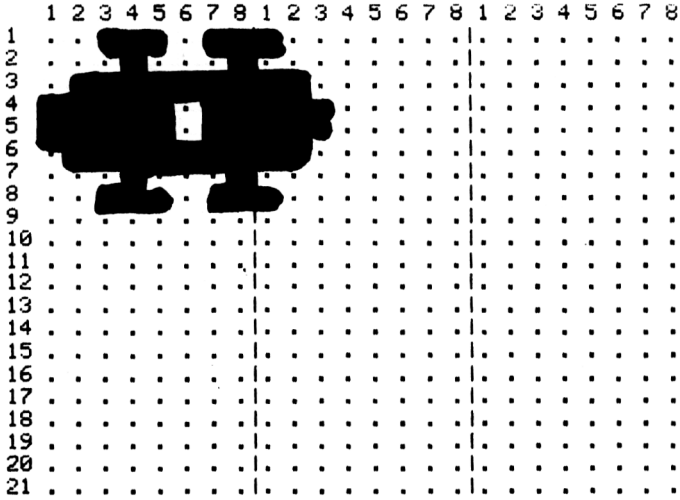


1	0	56	0
2	0	68	0
3	0	130	0
4	1	1	0
5	1	1	0
6	1	1	0
7	1	131	0
8	1	199	0
9	1	255	0
10	1	255	0
11	1	255	0
12	1	255	0
13	1	255	0
14	1	255	0
15	1	255	0
16	1	255	0
17	1	255	0
18	1	255	0
19	0	254	0
20	0	124	0
21	0	56	0

Course automobile

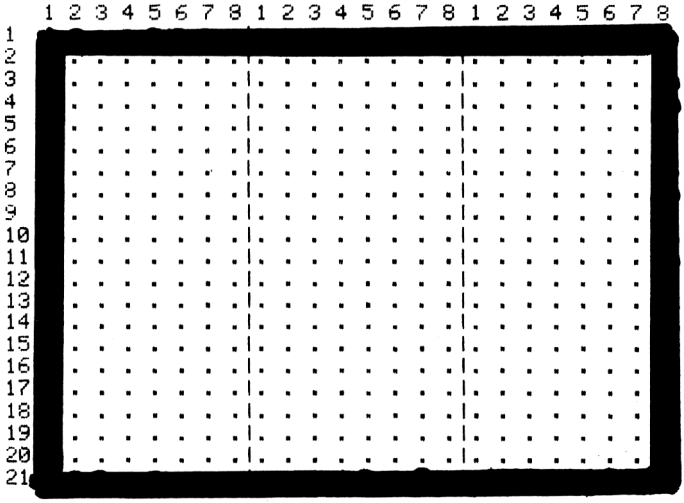


1	24	0	0
2	60	0	0
3	189	0	0
4	255	0	0
5	189	0	0
6	36	0	0
7	189	0	0
8	255	0	0
9	189	0	0
10	60	0	0
11	24	0	0
12	0	0	0
13	0	0	0
14	0	0	0
15	0	0	0
16	0	0	0
17	0	0	0
18	0	0	0
19	0	0	0
20	0	0	0
21	0	0	0



1	59	128	0
2	17	0	0
3	127	192	0
4	251	224	0
5	251	224	0
6	127	192	0
7	17	0	0
8	59	128	0
9	0	0	0
10	0	0	0
11	0	0	0
12	0	0	0
13	0	0	0
14	0	0	0
15	0	0	0
16	0	0	0
17	0	0	0
18	0	0	0
19	0	0	0
20	0	0	0
21	0	0	0

Puces savantes



255	255	255
128	0	1
128	0	1
128	0	1
128	0	1
128	0	1
128	0	1
128	0	1
128	0	1
128	0	1
128	0	1
128	0	1
128	0	1
128	0	1
128	0	1
128	0	1
128	0	1
128	0	1
128	0	1
128	0	1
255	255	255

2. Caractères spéciaux

clavier	couleur	représentation
[CTRL] [1]	NOIR	
[CTRL] [2]	BLANC	
[CTRL] [3]	ROUGE	
[CTRL] [4]	TURQUOISE	
[CTRL] [5]	VIOLET	
[CTRL] [6]	VERT	
[CTRL] [7]	BLEU	
[CTRL] [8]	JAUNE	
[C=] [1]	ORANGE	
[C=] [2]	MARRON	
[C=] [3]	ROUGE CLAIR	
[C=] [4]	GRIS 1	
[C=] [5]	GRIS 2	
[C=] [6]	VERT CLAIR	
[C=] [7]	BLEU CLAIR	
[C=] [8]	GRIS 3	

caractères de mise en page

clavier	fonction	représentation
[CLR/HOME]	retour du curseur	
[SHIFT] [CLR/HOME]	effacement de l'écran	
[CRSR]/GAUCHE	déplacement à gauche	
[CRSR]/DROIT	déplacement à droite	
[CRSR]/BAS	déplacement en bas	
[CRSR]/HAUT	déplacement en haut	
[CTRL] [ON]	mise en mode video inverse	
[CTRL] [OFF]	retour mode normal	

LA BIBLIOTHÈQUE EDIMICRO

- *Collection « Guides microordinateurs »*

de Merly	**GUIDE DE L'APPLE**
	Tome 1 **L'APPLE standard**
	Tome 2 **Les extensions**
	Tome 3 **Les applications**
	Tome 4 **Les langages**
Bayvejiel	**GUIDE DE L'ORIC**
Bieber, Perbost, Renucci	**GUIDE DU TO 7**
de Merly	**Ordinateur familial : que choisir ?**

- *Collection « Jeux »*

Chane-Hune, Darbois	**JEUX SUR ORIC**
Perbost, Renucci	**JEUX SUR TO 7**
Fleurier, Meiller	**Jeux sur Commodore 64 : adresse et hasard**
	Jeux sur Commodore 64 : action et réflexion

- *Collection « Progiciels »*

Bonnet, Dinh	**MULTIPLAN SUR APPLE Exercices de Gestion**
Bonnet, Dinh	**MULTIPLAN SUR IBM PC Exercices de Gestion**

- *Ouvrages de base*

Viguier	**Premiers pas en programmation** sur ORIC
Gaucherand, Lamoitier	**Fichiers en Basic par l'exemple**

Pour tout renseignement et catalogue gratuit :
F.D.S. Edimicro, 121-127, avenue d'Italie — 75013 Paris.

Achevé d'imprimer en février 1984
sur les presses de l'imprimerie Laballery et Cie
58500 Clamecy
Dépôt légal : février 1984
No d'imprimeur : 401043

JEUX SUR COMMODORE 64

Jeux d'adresse et de hasard

Philippe FLEURIER
Christian MEILLER

12 jeux d'adresse et de hasard, prêts à l'emploi, pour votre Commodore 64

- Course d'escargots
- Jeu du 21
- Balle au pied
- Squash
- Casse-briques
- Clavier
- Trampoline
- Soucoupes
- Tir sur cible
- Poursuite
- Course automobile
- Puces savantes

Chaque jeu est présenté en détail: **organigramme, étude ligne à ligne, liste des instructions du programme.**

Cet ouvrage s'adresse à tous les possesseurs et utilisateurs du Commodore 64. Les premiers chapitres vous indiquent **comment concevoir un programme BASIC** et vous décrivent les principales techniques de programmation.

Vous pourrez donc réaliser vos propres programmes de jeux, en mettant en œuvre les étonnantes possiblités graphiques et sonores de votre ordinateur.

Dans la même série : **JEUX D'ACTION ET DE RÉFLEXION SUR COMMODORE 64.**

Edimicro

121-127, Avenue d'Italie 75013 Paris

ISBN 2-904457-09-7

PRIX 88 F